University of New Orleans

ScholarWorks@UNO

University of New Orleans Theses and Dissertations

Dissertations and Theses

8-6-2009

Vehicle detection and tracking using wireless sensors and video cameras

Sowmya Bandarupalli
University of New Orleans

Vehicle detection and tracking using wireless sensors and video cameras

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
In partial fulfillment of the
Requirements for the degree of

Master of Science
in
Engineering

by

Sowmya Bandarupalli

August 2009

Acknowledgments

I would like to take this opportunity to thank Dr. X. Rong Li for taking me as his graduate student and for his patience and support, when I took way longer than expected to finish. I would also like to thank Dr. Huimin Chen for his advice, support and guidance. This work would not have been possible without his help and direction. I would like to thank Dr. Dimitrios Charalampidis for accepting to serve on my thesis committee. I would also like to thank Dr. Vesselin Jilkov and Dr. Rasheed Azzam for their support and guidance.

I would like to thank all the members of the Information and Systems Laboratory for sharing their knowledge and research experiences during the weekly seminars. It is very encouraging to be a part of such a motivated and hardworking group. I would like to thank Ashwini Amara for her time and valuable inputs. This research is supported in part by the Army Research Office and Department of Defense.

I thank my husband, Naveen who stood beside me and encouraged me constantly. Thank you for supporting me and taking care of everything. And my thanks go to my lovely daughter, Sanjana for her cooperation. I would like to thank my parents and in-laws who have supported me always. Above all, I would like to thank my Master, my spiritual guide for his blessings.

Table of Contents

Table of Figures v
Abstract vii
Chapter 1 – Introduction 1
1.1 Introduction to Target Surveillance 1
1.2 Outline of the thesis 5
Chapter 2 – Overview of the Surveillance Testbed 7
2.1 Introduction to the Surveillance Testbed 7
2.2 Components of the Testbed 10
2.3 Overview of Vehicle detection and tracking 11
Chapter 3 – Vehicle detection using wireless sensors 13
3.1 Overview of wireless sensor networks 13
3.2 Applications of Wireless Sensor Networks 13
3.3 Vehicle detection using wireless sensors 15
3.3.1 Micaz Mote – Kit 2400 15
3.3.2 MIB510 Programming board 17
3.3.3 Functionality of the acoustic and light Sensors 17
3.3.4 Supporting hardware and software 18
3.4 Hardware setup 19
3.4.1 Hardware Installation 19
3.4.2 Preparing the motes 20
3.4.3 Installing TinyOS applications onto a mote 20
3.4.4 Testing the motes 21
3.5 Data logging 24
3.5.1 Mote-View 24
3.5.2 PostgreSQL 8.0 25
3.5.3 SerialForwarder GUI 25
3.5.4 Xlisten 26
3.6 Sensor data monitoring using Mote-View 27

3.7 Sensor data retrieval 29
3.7.1 Using PostgreSQL and SQL commands 29
3.7.2 Using Xlisten and Matlab 30
3.8 Assumptions and limitations of data collection using wireless sensors 30
3.9 Sensor data processing for vehicle detection 31
3.10 Observations 35
Chapter 4 –Video based Vehicle Tracking 37
4.1 Background Information 37
4.2 Camera model 39
4.3 Overview of Camera calibration 41
4.3.1 Camera calibration matrix / Internal parameters matrix 42
4.3.2 External parameters matrix` 42
4.4 Different methods of estimating camera parameters 43
4.5 Preprocessing techniques 44
4.5.1 Processing of video files in Matlab 44
4.5.2 Vehicle extraction from images 45
4.5.3 Calculation of vehicle centroid 46
Chapter 5 –Experimental Study 50
5.1 Experimental setup 50
5.2 2-D to 3-D mapping 50
5. 3 Self calibration procedure 51
5.4 Calculation of 3-D world coordinates 57
5.5 Camera calibration using non-linear minimization techniques (fminsearch) 59
5.6 Some of the challenges during the experimental study 62
5. 7 Vehicle trajectory from multi-cameras 63
5. 8 Vehicle trajectory using wireless sensors 65
5. 9 Integration of vehicle detection and tracking 66
5. 10 Types of errors in a typical surveillance system 68
Chapter 6 - Conclusions and future work 71
Bibliography 74
Vita 76

Table of Figures

Figure 1: Testbed setup for Vehicle detection and tracking 9

Figure 2: Overview of vehicle detection and tracking 11

Figure 3: Micaz mote 16

Figure 4: MTS310CA multipurpose sensor board 16

Figure 5: MIB510 programming board 17

Figure 6: Micaz plugged to MIB510 21

Figure 7: Serial Forwarder GUI 26

Figure 8: *Mote-View* window illustrating the charts feature 28

Figure 9: Light sensor measurements 33

Figure 10: Acoustic sensor measurements 34

Figure 11: Camera perspective model 40

Figure 12: Extraction of an object from the background 48

Figure 13: Plots of vehicle centroid from camera 1 53

Figure 14: Plots of vehicle centroid from camera 2 53

Figure 15: Plot of vehicle centroids from camera 3 53

Figure 16: Vehicle trajectory from 3 cameras in the world coordinates - onward direction 61

Figure 17: Vehicle trajectory from 3 cameras in the world coordinates - return direction 62

Figure 18: 2-D average vehicle trajectory from video cameras 64

Figure 19: 1-D average vehicle trajectory from video cameras 64

Figure 20: Vehicle trajectory using wireless sensors 66

Figure 21: Vehicle trajectory using sensors and cameras 67

Figure 22: Vehicle trajectories from sensors, cameras and known sensor locations 69

Figure 23: Estimated error between vehicle locations from sensors and cameras 70

Abstract

This thesis presents the development of a surveillance testbed using wireless sensors and video cameras for vehicle detection and tracking. The experimental study includes testbed design and discusses some of the implementation issues in using wireless sensors and video cameras for a practical application. A group of sensor devices equipped with light sensors are used to detect and localize the position of moving vehicle. Background subtraction method is used to detect the moving vehicle from the video sequences. Vehicle centroid is calculated in each frame. A non-linear minimization method is used to estimate the perspective transformation which project 3D points to 2D image points. Vehicle location estimates from three cameras are fused to form a single trajectory representing the vehicle motion. Experimental results using both sensors and cameras are presented. Average error between vehicle location estimates from the cameras and the wireless sensors is around 0.5ft.

Keywords: Surveillance testbed, vehicle detection and tracking, camera calibration, perspective projection model, wireless sensor measurements, Micaz, MoteView. Nonlinear minimization, fminsearch, estimation of camera parameters, azimuth and elevation angles.

Chapter 1 – Introduction

1.1 Introduction to Target Surveillance

Surveillance is the monitoring of behavior or watching-over of people or objects of interest. In recent years, we have witnessed a rapid growth in the research and development of various surveillance methods using multimodal sensors, including video, audio, thermal, vibration, and various kinds of other sensors, in both civilian and military applications [1]. The objective of a surveillance system is to detect, classify, track, recognize and interpret behavior of objects of interest in the designated area. Surveillance systems can be categorized based on the type of devices it uses. Some representative surveillance applications include:

- Environmental, patient and habitat monitoring: Wireless sensors are popularly used due to the ease of deployment and monitoring. For applications such as habitat and environmental monitoring, deploying a large number of wireless sensors in a given coverage area seems to be effective due to the low cost and flexibility of data collection. On the other hand, a large amount of sensory data requires a complex data processing algorithm to achieve the surveillance objective.

- Conventional radar and camera based surveillance systems are generally built using wired devices with relatively high sensing accuracy and a durable power supply. However, due to the high cost and physical/geometric restrictions on the deployment, these sensing devices have to be more or less fixed at certain locations. Therefore, data collected by them are usually limited in spatial coverage and non-adaptive to the environmental changes.

- Surveillance systems with both wired and wireless devices: In regard to the pros and cons of the wired and wireless sensing devices, the integration of information obtained from both types of sensors with multiple modalities is expected to improve the object detection and tracking accuracy. For a practical design of the surveillance system, virtual reality, or smart videoconferencing system, a natural choice is to have a combination of fixed (mostly wired) devices which persistently cover the region of interest and wireless sensors which can provide more detailed information as needed.

Combining the results of multiple sensing devices can provide more accurate information than using a single device; this allows either improved accuracy from existing devices or the same performance from smaller or cheaper sensors.

In target surveillance, the network of sensors will have to cooperate to perform scene monitoring and continuous target tracking over a large area that cannot be viewed continuously by a single sensor alone. Cooperative multi-sensor surveillance will significantly enhance situational awareness by providing complete and continuous coverage of target movements in the coverage area. Information in the form of estimated target locations and appearances can be gathered from different sensors of possibly different sensing modalities and redundant data needs to be correlated and merged for better results.

Surveillance applications such as target detection, tracking classification, activity understanding has attracted much attention. Active research is being done for vehicle detection and tracking using wireless sensors. The idea of deploying wireless sensors to monitor the behavior of the target is not new. Different modalities of the sensors are being used to improve the accuracy of vehicle detection or tracking. Multimodal sensing has attracted much attention in solving a wide range of problems. In [2], both the acoustic and magnetic sensor measurements are processed for vehicle detection. Individually, the acoustic and magnetic sensors can detect the targets regardless of the bearing with low power consumption.

Video based surveillance systems typically monitor the behavior of the target in the designated area using single or multiple cameras. These systems require the development of image processing and computer vision algorithms to detect, locate and track a moving target. The use of multiple cameras enables the surveillance of greater areas and the use of the redundant information (provided by the videos from different viewpoints) can help overcome some of the known limitations of single camera systems, such as scene occlusions. These limitations are mainly due to two types of problems. There are certain areas that cannot be processed by the cameras particularly in cluttered scenes and can be solved by using multiple cameras and sensors. The second class of problems is due to the limited resolution of cameras. An interesting solution to these problems could be using simple but effective specialized sensors with different modalities to solve the specific problems of the vision systems. In this way, vision would still provide high-level information, and low-level sensors would assure higher accuracy.

In order to establish a correspondence among the information from different views, it is necessary to determine a mapping function that relates the location of a pixel in one view with that of the same pixel in another view. This mapping function is known as Homography. In [3], an effective method is presented to resolve occlusions and to track objects between overlapping and non-overlapping camera views using the homographic relations between each camera view.

Active research on vehicle detection and tracking using multimodal sensors has recently become an important application in target surveillance. To intelligently fuse information from different modalities, novel strategies for detection and data association have to be developed to exploit the multimodal information. In the literature, one finds that fusion of acoustic and video modalities has been applied to problems such as tracking of humans under surveillance and smart videoconferencing. The success of the fusion strategy is mainly because each modality may compensate for the weaknesses of the other or can provide additional information. In surveillance applications, different types of sensors, such as video and acoustic sensors, provide distinct observations of ongoing activities. In [4], a fusion framework is presented using video and acoustic sensors for vehicle detection and tracking. Both video cameras and acoustic arrays have been used as sensing devices to estimate these parameters. A computational framework for joint audio-visual vehicle detection and tracking using Markov Chain Monte Carlo (MCMC) techniques is presented. Improved tracking performance is reported by fusing the empirical posterior probability density functions obtained using both types of sensing modalities. To enhance video surveillance systems, multi-modal sensor integration can be a successful strategy. In [5], a computer vision system capable of detecting and tracking people from multiple cameras is integrated with a wireless sensor network mounting Passive Infra Red (PIR) sensors. The reported results demonstrate that integration of PIR sensors and cameras can significantly increase the accuracy.

To understand the pros and cons of both wired and wireless sensing devices, we worked with a set of wireless sensors and few cameras. From the documentations and manuals provided with the sensors, we were able to configure, program and test the wireless sensors before the actual data acquisition could begin. Having hands-on experience provided in-depth understanding about the basic functionality as well as the limitations. Knowledge about different sensing devices helped us design of the testbed, fuse the data from multi-sensor modalities and improve the accuracy of the vehicle detection and tracking system.

This thesis reports the development of a testbed integrating a wireless network of Micaz mote sensors and a set of three video cameras. We have developed a target surveillance testbed for small-scale illustration of the automated vehicle detection and tracking using various algorithms. In the case of partially overlapped cameras, the best choice is to exploit the geometrical information. These geometry based methods can be classified into calibrated and un-calibrated approaches. Each camera processes the scene and obtains a set of tracks. The objective of this thesis is to combine the videos obtained from the cameras with the data provided by the wireless sensor network to effectively detect and track a moving vehicle. In our experimental study, we used the wireless sensors to detect the vehicle. The high-resolution video cameras were used to generate the vehicle trajectory from the video sequences obtained. The testbed provides a mechanism for performance evaluation of algorithms developed in different research areas including target information processing, integrated sensing and data fusion, image processing and target tracking. Video surveillance technologies are generally based on data processing and analysis for moving object detection, region localization, object classification, object identification, tracking, human motion analysis, and activity understanding.

Part of the research results in this thesis has been published at the International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities (TridentCom 2007). As one of the coauthors of the TridentCom paper [6], my contribution includes implementing the sensor data processing and camera calibration algorithms. During the initial phases of the surveillance testbed, a PhD student in our team, Ming Yang, provided much insight about surveillance systems in general and different data fusion techniques used to detect and track a vehicle. My contributions in different phases of the testbed development include:

- Install the required hardware and software components, understand the functionality of the different modalities of the micaz motes, and program the mote sensors with applications required for the experiment.
- Implement sensor data acquisition, monitor and retrieve the sensor data from the database, and develop an algorithm for sensor data processing to detect the presence of the vehicle.
- Address the basic challenges in vehicle detection and tracking using multiple wireless sensors and video cameras in our testbed and the techniques we have adapted to overcome the challenges.
- Understand the basics of Camera perspective model, camera calibration and different methods of calibrating a camera. Develop an algorithm to preprocess the video sequences and to calibrate the camera using the self-calibration technique.
- Estimate the camera parameters using *fminsearch* a nonlinear minimization method that find a solution by minimizing the errors between the actual image coordinates of the reference points and the estimated image coordinates.
- Summing it up, work on the development of a surveillance testbed to perform small scale vehicle detection and tracking using wireless sensors and video cameras.

1.2 Outline of the thesis

Chapter 2 presents the overall structure of the surveillance testbed. Background information about target surveillance and the design of the testbed are discussed. The basic components of the testbed are discussed.

Chapter 3 gives an overview of vehicle detection using wireless sensors. Some of the applications of wireless sensor networks are discussed. The micaz development kit, the programming board, etc. are described in detail. The TinyOS operating system and the nesC programming language are also explained related to the collection and processing of the sensory data. Vehicle detection using wireless sensors is described in detail. Basic functionality of the light and acoustic sensors is discussed. Some issues encountered during the data acquisition process are discussed along with the solutions. The data log, monitoring and retrieval procedures are discussed.

Chapter 4 presents an overview of the vision based vehicle tracking module. Different methods of vehicle tracking are discussed. A camera model is discussed with an emphasis on the perspective projection model. An overview of camera calibration and different methods of calibrating a camera are discussed. Detailed information about camera parameters is presented, and different ways of estimating the camera calibration matrix and the camera pose are described. Preprocessing techniques such as background subtraction and calculation of the vehicle centroid are discussed.

In Chapter 5, the experimental setup and the basic challenges faced during the experiment are discussed. We have a set of points whose coordinates in three dimensions are known, and whose locations in two dimensions are also known. In addition to these points, we also know that the view is a perspective projection, and therefore the transformation of the three-dimensional coordinates of a point to the two-dimensional coordinates can be expressed compactly as a 3 x 4 transformation matrix using homogeneous coordinates. An algorithm to obtain the transformation matrix and perform the 2-D to 3-D mapping is outlined. The camera parameters are estimated using a method of nonlinear minimization of the errors between the actual image coordinates of the reference points and the estimated image coordinates. The parameters estimated using the self calibration procedure is used as the initial values to the nonlinear method. One of the challenges is how to track the vehicle using multiple sensing devices and how to fuse all the estimates. This chapter presents the integration of vehicle detection by the wireless sensors and vehicle tracking by the video cameras. Different types of errors in a typical surveillance system are presented. Error between the estimated vehicle locations with respect to the cameras and sensors is computed.

Chapter 6 presents conclusions and also discusses possible future work to improve the accuracy of the vehicle tracking with the testbed at different levels.

Chapter 2 – Overview of the Surveillance Testbed

2.1 Introduction to the Surveillance Testbed

One major surveillance task is to detect and track moving targets (e.g., vehicles, people). Surveillance systems track the targets and also detect and classify. Integration of the multi-sensor and multi-camera output to obtain an evolving, scene representation of the target is an important task. This is a typical problem of information fusion. In simple words, *information fusion* is to combine information (data, decisions, estimates, etc.) from multiple sources (sensors or data processing nodes, etc.) to achieve better inference than could be achieved by the use of a single source [6]. Using multiple cameras to track a target has several advantages. Fast moving targets can often still be tracked. If the target moves fast enough to evade one camera, it is possible that the other cameras could still keep the target in view.

Deploying a large number of wireless sensor nodes in the designated area is usually not a good choice. It is not only inefficient, but also hard to achieve accurate and robust performance. The advantages such as low cost and low energy consumption are outweighed by the total cost for collaborative sensing and data processing. At the same time, we cannot just use wired devices due to the high cost and the physical/geometric restrictions on the deployment. In regard to the pros and cons of both approaches, a viable choice of system design would be to have a combination of fixed (wired) devices which persistently cover the region of interest and wireless sensors which can provide more detailed information as needed.

A system consisting of a set of distributed sensor platforms is capable of observing and interacting with a larger region of the coverage area. A greater amount of information about the environment can be obtained with a reduced uncertainty through cooperation and sharing of the environmental knowledge obtained by fusion of multiple viewpoints. In our experimental study, we use the wireless sensors to detect the vehicle and the video cameras to track the vehicle. Video surveillance technologies are generally based on data processing or analysis techniques for moving object detection, region localization, object classification, object identification, tracking, human motion analysis, and activity understanding. These techniques touch on many of the core theories and topics in computer vision, pattern analysis, and artificial intelligence. There is a standard set of techniques for video surveillance systems that use background modeling or frame differencing to detect moving objects of interest or feature tracking from frame to frame.

The basic techniques for interpreting video and extracting information from it have received a significant amount of attention. Video based image tracking aims to develop the object trajectories over time from the video sequences obtained. From the obtained trajectories, and, with the knowledge of camera pinhole model and other parameters, position of the object can be determined in the scene with reference to a three-dimensional (3-D) world coordinate system. This 3-D point refers to the vehicle location in the real world. We evaluate the performance by comparing the vehicle trajectory obtained using the video cameras to that obtained from the wireless sensors. We have developed a target surveillance testbed for a small-scale illustration of the automated vehicle detection and tracking using various algorithms. The equipment used to develop the testbed system includes:

- Remote control vehicle
- Two desktop computers and two laptops: for collecting data and local processing.
- Two work stations used as fusion centers.
- Wireless sensor nodes and developing kits (Micaz Mote- Kit2400): the programming board is connected to a desktop via a serial-USB converter.
- Overhead cameras and a surveillance center (currently built on a high performance desktop).
- Wireless routers (802.11b/g).

The testbed setup for this particular experiment involves both the wireless sensors as well as the video cameras. The experiment was carried out in a dark room with a coverage area of approximately 20ft×4ft. Ten sensors are placed equally along both sides of the coverage area. A remotely controlled vehicle with two sensors and a flash light onboard was used. The remotely-controlled vehicle will be monitored by different sensors. Three video cameras and a digital camera are used to provide the scenes of surveillance area and possibly the position information of the vehicle. The video cameras are placed at 0ft, 15ft and 23ft respectively and at an elevation of approximately 3ft along x-axis. The video cameras are mounted on fixed locations, recording video streams and directly communicating to a desktop by a multi-channel wireless receiver. The laptop enables the processing of the sensor programming board which is collecting sensing data from each Micaz mote. The Micaz motes with acoustic and light sensor functionalities are used as low resolution sensors deployed in the experimental area.

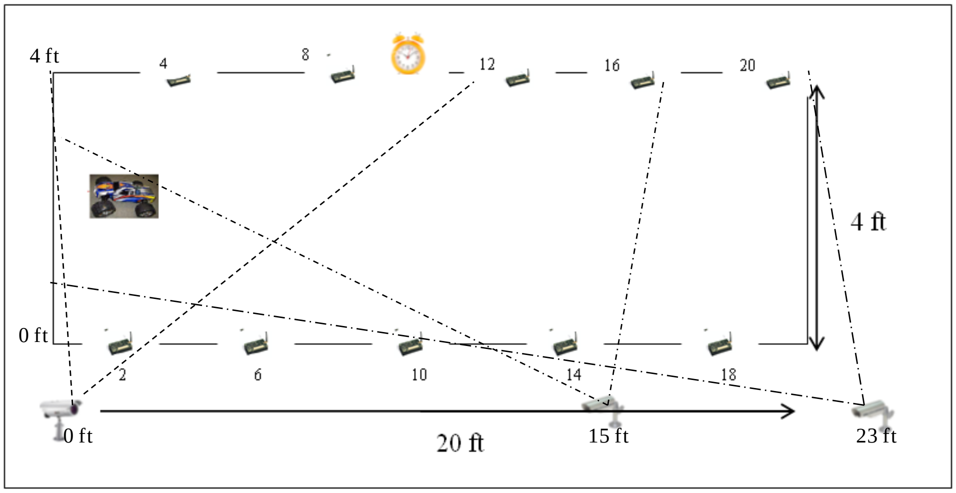

Figure 1: Testbed setup for Vehicle detection and tracking

The remotely-controlled vehicle is be monitored by different sensors. The Micaz motes with acoustic/light sensors are used as low resolution sensors deployed in the experimental area. In order to check the sensitivity and accuracy of the wireless sensors in terms of missed detections, false alarms and the actual detection of the vehicle, different scenarios were tested.

- Vehicle moving slowly in a straight line with two sensors and a flash light mounted
    - Flash light pointing in the same directions in both onward and return paths.
    - Flash light pointing in different directions in both onward and return paths.
    - Vehicle making a turn in the coverage area.
- Vehicle moving fast in a straight line with two sensors and a flash light mounted

- Flash light pointing in the same directions in both onward and return paths.
- Flash light pointing in different directions in both onward and return paths.
- Vehicle making a turn in the coverage area.

Inorder to check the sensitivity of the sensors, in the first scenario, we placed the flashlight on the vehicle pointing to the same direction in both onward and return paths. Since we have the ten sensors placed equally on both sides of the coverage area, we wanted to test how many of the sensors would actually detect the change in the light conditions. In the second scenario, since the direction of the flashlight is changed in the onward and return paths, all the sensors should be able to detect the changes in the light conditions. Similarly, the first two scenarios are repeated with the vehicle's speed increased from 1m/s to approximately 3m/s. Sensor data is recorded into the database as the vehicle moves along the sensing area. We also checked the measurements of the 2-axis accelerometers on board the vehicle for any indication of the vehicle's motion and direction, especially when the vehicle was making a turn. Corresponding light, voltage/temperature, acoustic, accelerometer, magnetometer readings are logged into the database for all the time instants. The position of the laser pointer was adjusted on the vehicle in such a way that the light sensors can easily sense the laser beam.

2.2 Components of the Testbed

The key components of the testbed are:

- *Micaz Mote Kit 2400:* The Micaz development platform, Mote-Kit 2400, is a comprehensive 8 node kit for commercial Mote development from Crossbow Technology, Inc [7]. The Micaz motes are designed for deeply embedded sensor networks.

- The 2400 series kit has all of the components needed to develop, test and implement a wireless sensor network. The kit includes: MTS310CA sensor board with multiple sending modalities. The sensor boards can measure light, temperature, barometric pressure, acceleration/seismic and acoustic sounds.

- *BU 581SRW Sony CCD Bullet camera:* The smallest outdoor color bullet camera using a Sharp 1/3" CCD. The camera features 480 TV Lines of high resolution and 1.0 LUX of excellent low light sensitivity. It is easy to exchange the lens with a variety of angle of depending on the requirement of the application [8].

2.3 Overview of Vehicle detection and tracking

This section explains steps in the vehicle detection and tracking using wireless sensors and video cameras. Figure 2 shows different modules in the form of a block diagram.

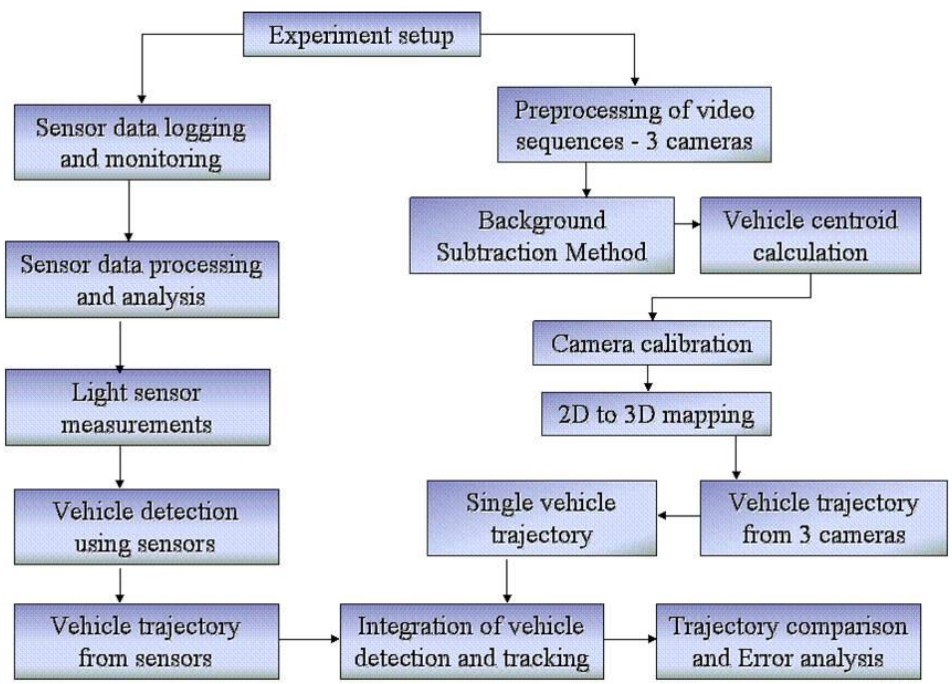

Figure 2: Overview of vehicle detection and tracking using wireless sensors and video cameras.

Experimental setup: This section deals with the details of the experimental setup and the key components of the testbed. The testbed setup involves both the wireless sensors and the video cameras.

Sensor data logging and monitoring: Functional details of light and acoustic sensors are discussed. This section also describes the software needed for sensor data logging and steps to installing TinyOS applications onto a mote sensor. Different methods of retrieving sensor data measurements are discussed.

Vehicle detection using sensor measurements: This section discusses the sensor data processing algorithms and detection of the vehicle using sensor data measurements.

Preprocessing of video sequences: The preprocessing techniques include conversion of a video sequences into set of images, background subtraction method to extract the target from an image and calculation of the vehicle centroid.

Camera calibration: Camera calibration is an important part of most computer vision systems. Camera's intrinsic parameters (e.g., the focal length) and its position in the world coordinate system are not known in advance. All these parameters are determined using geometric primitives commonly found in traffic scenes. Using these parameters, ground plane rectification can be done. If a pixel in the image appears on the ground plane, its 3-D coordinates can be found in the world reference frame.

Integration of camera and sensor data measurements: In our experiment, the wireless sensors were used for vehicle detection and video cameras were used for tracking. Tracking is considered a process of data association that links the detection results from individual frames to a temporal trajectory. One of the main challenges we faced during the experimental analysis was to fuse the data obtained (vehicle location estimates) from both wireless sensors and the three cameras to obtain a meaningful representation of the vehicle's motion. The method that we adopted will be discussed in Chapter 5.

Error Analysis: Video data is compared against the sensor measurements used for vehicle detection. The responses of vehicle detectors, i.e., the sensors are accumulated over time to obtain the trajectory of the detected vehicle locations. The detected vehicle is tracked in the video frames with the help of perspective projection model and the camera parameters. We estimate the vehicle motion by calculating the centroid of the vehicle in each of the frame, and then find its corresponding location in the real world.

Chapter 3 – Vehicle detection using wireless sensors

3.1 Overview of wireless sensor networks

Wireless sensor networks are comprised of a large number of small sensing devices commonly known as motes. The motes have a low clock rate processor on board as well as a small amount of memory [9]. They also have some form of a specific or multiple functionalities attached to them in order to monitor physical properties. These sensors can be built directly into the motes main board or can come as add on boards which can be connected in some way to the mote. Collectively these motes are able to form themselves into autonomous ad-hoc networks using a variety of communication mediums. The most common medium is radio frequency communication. These networks of motes are able to collect, process and store data from sensors. They are also capable of sharing data with each other.

Wireless sensor networks are very versatile and can be used in many different application areas. The individual motes that form a sensor network are relatively inexpensive. They require no maintenance once deployed as they automatically form a wireless ad-hoc network which is completely autonomous. Recent examples of applications developed for these networks include tracking military vehicles and monitoring forest fires. To support these applications the TinyOS operating system has been developed to control the operation of the mote devices. Among other software products for wireless sensor networks, TinyOS provides a customizable networking stack which includes message passing functions. This allows motes to form an ad-hoc network in order to communicate with other motes and possibly other types of devices.

3.2 Applications of Wireless Sensor Networks

The applications using wireless sensor networks can be classified into six major categories such as Military, Environmental, Habitat monitoring, Health, Home and Office, among others [9]. Some applications are briefly listed below under each major category.

Military Applications

Wireless sensor networks can be an integral part of military command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting systems. Since sensor networks are based on the dense deployment of disposable and low-cost sensor nodes, destruction of some nodes by hostile actions does not affect a military operation which makes sensor networks concept a better approach for battlefields. Some of the military applications of sensor networks are battlefield surveillance, targeting, battle damage assessment, and nuclear, biological and chemical attack detection.

Environmental Applications

Some environmental applications of sensor networks include tracking the movements of birds, animals, and insects; monitoring environmental conditions that affect crops and livestock, chemical/ biological detection, forest fire detection, flood detection and pollution study.

Habitat Monitoring Applications

Sensor networks are being deployed in natural parks and wildlife reserves to closely monitor and aggregate data from animal and plant life. Great Duck Island System is one such application for Habitat Monitoring [10] where the Mica motes are deployed using an Atmega 103 microcontroller running at 4M Hz, 916M Hz radio from RF monolithic to provide bidirectional communication at 40kbps and a pair of AA batteries. 32 motes were placed at area of interest. These motes transmit sensor data to a gateway, which is responsible for forwarding the data to a remote base station.

Health Applications

Some of the health applications for sensor networks are providing interfaces for the disabled; integrated patient monitoring diagnostics; drug administration in hospitals; tele-monitoring of human physiological data; and tracking and monitoring doctors and patients inside a hospital. Some other similar applications include glucose level monitors, organ monitors, cancer detectors and general health monitors. The idea of embedding wireless biomedical sensors inside human body is promising, although many additional challenges exist: the system must be extremely safe and reliable and must require minimal maintenance.

Home and office Applications

Sensors are envisioned to be ubiquitous, integrating themselves to all household appliances. Such devices are connected to actuators which take an action when the environment changes to a particular state. End users could communicate with these devices making control decisions remotely. Some of the important applications could be Smart Homes where sensors make intelligent decisions such as controlling the room temperature, turning on the lights etc and Environmental control in office buildings where a distributed sensor network can be used to control the air flow and temperature.

3.3 Vehicle detection using wireless sensors

Wireless sensors are integrated with video cameras to develop a target surveillance testbed for small scale illustration of vehicle detection and tracking. In this chapter we describe the experimental work for vehicle detection using sensor node data. Both light and acoustic signals are processed for vehicle detection. The sensor nodes used in our vehicle detection experiments are the Micaz motes, a part of the Micaz development platform, Mote-Kit 2400. The key components are described in detail along with the functionality.

3.3.1 Micaz Mote – Kit 2400

The Micaz development platform, Mote-Kit 2400, is a comprehensive 8 node kit for commercial Mote development from Crossbow Technology, Inc [7]. Being low-power, IEEE 802.15.4 compliant, offering 250kbps high data rate, and a ‘plug and play mote’ compatible to all the sensor boards and gateways, we have chosen the Micaz motes in our testbed. The Micaz motes are designed for deeply embedded sensor networks. Belonging to the Mica line of Cross-Bow, it features new Micaz (MPR2400CA) Processor/ Radio board. The Micaz has a 51-pin expansion connector that supports analog inputs, digital I/O, SPI and UART. The expansion connector works for light, temperature, RH, barometric pressure, acceleration/seismic, acoustic, magnetic, and other Crossbow sensor boards.

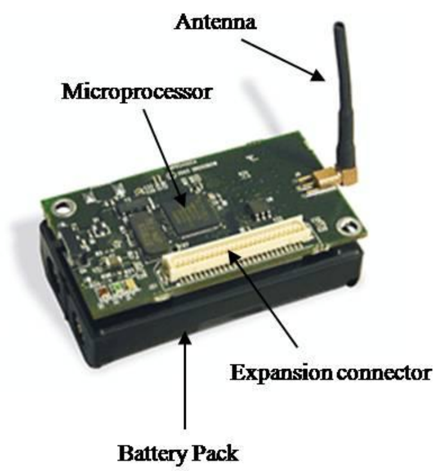

Figure 3: Micaz mote

For more details and specifications, please refer to the Micaz manual [11].

The 2400 series kit has all of the components needed to develop, test and implement a wireless sensor network. The kit includes: MTS310CA sensor board with multiple sending modalities as shown in Figure 4. The sensor boards can measure light, temperature and acoustic sounds and also has a sounder. We have used the light and acoustic modalities in our surveillance testbed. The basic components are: the mother board consisting of an Atmel 90LS8535 processor, 512KB SRAM, 8KB Flash RAM and a RF transceiver for wireless communication.

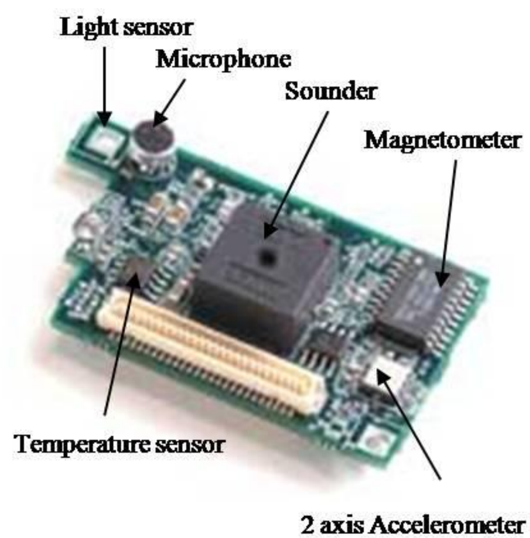

Figure 4: MTS310CA multipurpose sensor board

The Senor board consists of a 10-bit analog to digital converter, a Magnetomer(Honeywell HMC1002), a temperature sensor, a photo camera and an accelerometer sensor.

3.3.2 MIB510 Programming board

The MIB510 gateway allows for the aggregation of sensor network data on a PC. In addition to data transfer, the MIB510 also provides an RS-232 serial programming interface. With an onboard processor, it is capable of programming Micaz and Mica2DOT processor radio boards.

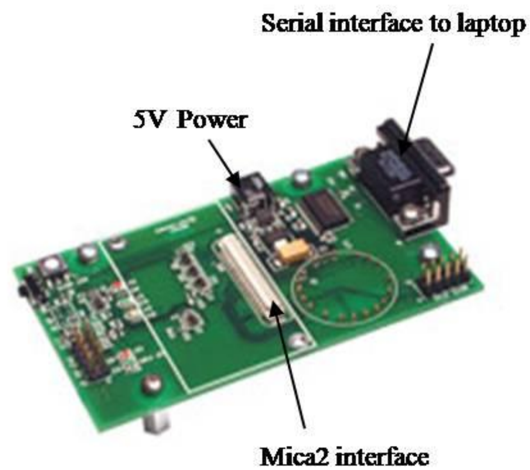

Figure 5: MIB510 programming board

It acts as a base station for wireless sensor networks via standard MICA2 processor radio board and supports serial port programming for all MICA hardware platforms. For more details and specifications, please refer to the Micaz manual [11].

3.3.3 Functionality of the acoustic and light Sensors

In vehicle detection module, the sensors we adopted are the light and acoustic sensors. Basic operating principles of the light and the acoustic sensors are presented [12].

Light Sensor:

The MTS310 sensor board has a light sensor and a thermistor. The light sensor is a simple CdSe photocell. A photoresistor or light dependent resistor or cadmium sulfide (CdS) cell is a resistor whose resistance decreases with increasing incident light intensity. It can also be referred to as a photoconductor. Based on the working principle of the light sensor, if the light from the vehicle is of high enough frequency, the resistance of the photocell is lowered thereby increasing the sensor measurements corresponding to the vehicle's proximity to the sensor. The collected sensor data is used to detect the vehicle's presence.

Acoustic Sensor:

Surveillance systems with acoustic sensing devices measure the acoustic energy or audible sounds produced by a variety of sources within a vehicle. The overall sound energy level increases when a vehicle passes the detection zone. The sensor node used was Micaz with sensor board MTS310. The acoustic sensor on the sensor board MTS310 is a Panasonic WM-62A microphone, which is an omni-directional condenser type microphone. A stretched metal diaphragm forms one plate of a capacitor. A metal disk placed close to the diaphragm acts as a backplate. When a sound wave excites the diaphragm, the capacitance between the two plates varies according to the variation in the sound pressure. The change in the capacitance generates an output proportional to the sound pressure, which is at an ultra low-frequency pressure variation. A high-frequency voltage (carrier) is applied across the plates and the acoustic sensor output signal is the modulated carrier. Vehicle detection from the acoustic sensor measurements is discussed in the subsequent sections.

3.3.4 Supporting hardware and software

TinyOS

- An open-source OS designed for wireless embedded sensor networks. Its component-based architecture enables rapid implementation while minimizing code size as required by the severe memory constraints inherent in sensor networks.

- Not an operating system ("OS") in the traditional sense; it is a programming framework for embedded systems and set of components that enable building an application-specific OS into each application.
- No file system, supports only static memory allocation. TinyOS has been built using nesC. It is built using components and interfaces.

nesC

The TinyOS system, libraries, and applications are written in nesC, a new language for programming structured component-based applications. The nesC language is primarily intended for embedded systems such as sensor networks. nesC has a C-like syntax, but supports the TinyOS concurrency model, as well as mechanisms for structuring, naming, and linking together software components into robust network embedded systems. The principal goal is to allow application designers to build components that can be easily composed into complete, concurrent systems, and yet perform extensive checking at compile time. Additional details about TinyOS and nesC can be found in the manual [13].

3.4 Hardware setup

Before the wireless sensors can be used for vehicle detection, the hardware setup and installation of the applications onto the motes has to be completed. After the installation is over, the motes have to be verified if the programs installed are working properly. Only after the hardware check is completed, the motes are ready for the experiment and data acquisition. This section will describe the steps required to setup, test and program the Micaz hardware. For further details about the hardware setup, please refer to the TinyOS tutorial on testing the hardware [14].

3.4.1 Hardware Installation

Switch Settings*:* The Micaz platform has a single slide switch which should always be in the OFF position while programming the motes. It is the power switch for the batteries. In the OFF position, the switch disconnects the batteries. During normal operation (not connected to the programmer), slide the switch to the ON position which will provide battery power to the mote. The MIB510 programmer has a single slide switch that affects how data is sent on the serial com port. It must be in the OFF position to both program and use the motes.

3.4.2 Preparing the motes

- *Install the appropriate batteries;* Attach the antennas to the Micaz motes if not already connected. Switch Micaz battery switch to "ON" position. As a first step, the MTS300 Sensor Board is attached to one of the Micaz motes by pressing its 51-pin connection socket with that of Micaz. Two AA batteries each are then placed in the motes, except in the mote that is to be attached with the MIB510 base station.

- *Setting up the base station:* Attach the Micaz mote labeled "Base_###_0" or "Surge_0" to the MIB510, the base station, where ### refers to the frequency band of the Mote (i.e., 2400 in our case). Supply power to the MIB510 with the AC wall power adaptor. One thing to note here is that when connecting the mote to the MIB510 base station, one tends to push the mote onto the MIB510 51-pin connector with less force since both the mote and the base station seem quiet fragile.

- Connect the MIB510 to the PC's serial port (or USB to serial or PCMCIA serial adaptor) with a straight-through RS-232 cable.

3.4.3 Installing TinyOS applications onto a mote

The programming tools also include a method of programming unique node addresses without having to edit the TinyOS source code directly. To set the node address/ID during program load, the general syntax for installing is:

make *<platform>* re|install,*<n> <programmer>,<port>*

where <programmer> *,<port>* name of the programmer, port ID or address or number of the host PC to which the programmer is attached,*,<n>* is an optional number (in decimal) to set the node ID or address, and *<platform>* is the type of Mote processor/radio hardware platform. Here is the command to assign a node address/ID of 38 to a MICA2 and the MIB510 programming board on the PC's COM1 serial port.

make <platform> install,38 mib510,<COM1>

The difference between install and reinstall is explained below.

install,*<n>*—compiles the application for the target platform, sets the node ID/address and programs the device (Mote).

reinstall,*<n>*—sets the node ID/address and downloads the pre-compiled program (into the Mote) only and does not recompile. This option is significantly faster.

- TinyOS application is programmed individually on all the motes. Open the cygwin window and point to the following location: ***C:/tinyos/cygwin/opt/tinyos-1.x/contrib/xbow/apps/***. Based on the type of mote being programmed, navigate to the corresponding folder under /apps.

- The first step is to install the TinyOS application on the base station using the command

make mica install, 0 mib510, com4,

Where '0' refers to the 'n^{th}' number of mote and mib510 refers to the programmer and com4 refers to the port number to which the programmer is connected to. The same procedure is repeated to program all the remaining motes.

3.4.4 Testing the motes

This section will describe how to test the Micaz mote hardware. Detailed information about testing the motes can be found in [14].

Figure 6: Micaz plugged to MIB510

Making MicaHWVerify

Open a ‘cygwin bash shell’ and type the following commands:

cd /opt/tinyos-1.x/apps/MicaHWVerify make mica2

At this point we now have a custom Micaz compatible program ready to be programmed into the mote.

Programming MicaHWVerify via the MIB510CA programmer

The previous step created an application ready for transferring into the Micaz. To write the firmware image to the Micaz chip, type the following command:

make mica reinstall mib510,/dev/ttyS0

Note**:** The above command is case sensitive. /dev/ttyS0 = com1 /dev/ttyS1=com2 /dev/ttyS2=com3 /dev/ttyS3=com4.

Hardware COMMS check

Now we know that the programming tools and the computer's parallel port are working. The next step is to verify the mote hardware. First, confirm that the LEDs are blinking like a binary counter. Next, connect the programming board to the serial port of the computer. This step will attempt to communicate with the Micaz. Make sure the slide-switches on both the programmer board and the Micaz are set to OFF. To read from the serial port, we provide a java tool called ***hardware_check.java***. It is located in the same directory. Build and run this tool. Type the following command:

make –f jmakefile

MOTECOM=serial@COM1:<baud rate> java hardware_check

Example, if COM1, the port is being used at 19.200 Kbaud to connect to the programming board. The commands listed above are case sensitive. It is helpful to perform this test on at least 2 Micaz nodes. This program checks the serial ID of the mote, the flash connectivity, the UART functionality and the external clock. If all status checks are positive, the hardware verification successful message will be printed on the PC screen. If any failure is reported on the monitor, it is better to replace the mote.

Radio Verification Test

This test requires 2 Micaz nodes. One unit will be used as the base root node and the second node will be used as the remote node. During the test, both the nodes will communicate with each other via the radio link.

Node #1: Take one of the nodes that has passed the 'Hardware COMMS Check', install the batteries and slide the switch to ON. The lights on the Micaz will start to blink, the node is now active!

Node#0: This node will be the root node and will stay connected to the programmer board. It must have some special 'base station' software programmed onto the Micaz in order for this test to operate correctly. To compile and load the base station software onto the Micaz, type the following commands:

cd /opt/tinyos-1.x/apps/TOSBase

make mica2

make mica2 reinstall mib510,/dev/ttyS0

Re-run the 'Hardware Comms Check'

cd /opt/tinyos-1.x/apps/MicaHWVerify

MOTECOM=serial@COM1:mica2 java hardware_check

The Node Serial ID report will be displayed. The ID of the remote node was displayed, not the Node ID of the Micaz plugged directly into the base station. The base station acted as a data conduit passing the serial data onto the radio network; the remote node responded to the data on the RF interface. If the remote mote is turned off or not functioning, it will return a message "Node transmission failure".

3.5 Data logging

3.5.1 Mote-View

Wireless sensor networks monitoring software developed by Crossbow Technology [15]. It is an interface ("client tier") between a user and a deployed network of wireless sensor providing the tools to simplify deployment and monitoring. It provides an easy means of logging wireless sensor data to a database, analyzing and plotting sensor readings. The setup looks as follows: all the sensor nodes are preloaded with *XMesh* software. Programming the nodes can be done from within the Mote-View application. Basically, the applications running on the nodes send sensor reading data to the gateway node which posts the readings to a database. Different applications for different node configurations (sensing equipment, processor, power modes and frequency bands) are included with the Mote-View software. It requires a full-fledged database running on the machine where the sensor data is stored. Depending on the type of mote being programmed, data is logged into corresponding table in the postgreSQL database. Mote-view connects to the remote Postgres database through a TCP/IP link.

Mote-View has a number of useful features for monitoring and understanding wireless sensor data. These features include:

- Historical and Real-Time Charting
- Topology Map and network visualization
- Data Export capability
- Printing the graph results
- Node programming with MoteConfig
- Command interface to sensor networks

Additional Software Requirements: For the application to run the following additional components are required:

- MoteConfig

- PostgreSQL 8.0 database service and postgreSQL ODBC driver
- Microsoft .NET 1.1 framework
- Surge-View

The installation files are included on the *Mote-View* installation CD.

3.5.2 PostgreSQL 8.0

- A powerful, open source relational database system known for reliability, data integrity, and correctness. Is highly scalable both in the sheer quantity of data it can manage and in the number of concurrent users it can accommodate [15].
- Runs on all major operating systems, including Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), and Windows.
- All the visualization tools in *Mote-View* require being connected to a database. This database can reside on the PC ("localhost"), a remote server, or a Stargate. The size of this database is bound by the storage available on the system.
- The installation of PostgreSQL will automatically install and configure a local *PostgreSQL* 8.0 database on the machine [15].

3.5.3 SerialForwarder GUI

- A program written in Java, it is used to read packet data from a computer's serial port and forward it over a server port connection, so that other programs can communicate with the sensor network via a sensor network gateway [16].
- It does not display the packet data itself, but rather updates the packet counters in the lower-right hand corner of the window.
- Once running, the serial forwarder listens for network client connections on a given TCP port (9001 is the default), and simply forwards TinyOS messages from the serial port to the network client connection, and vice versa. Multiple applications can connect to the

serial forwarder at once, and all of them will receive a copy of the messages from the sensor network.

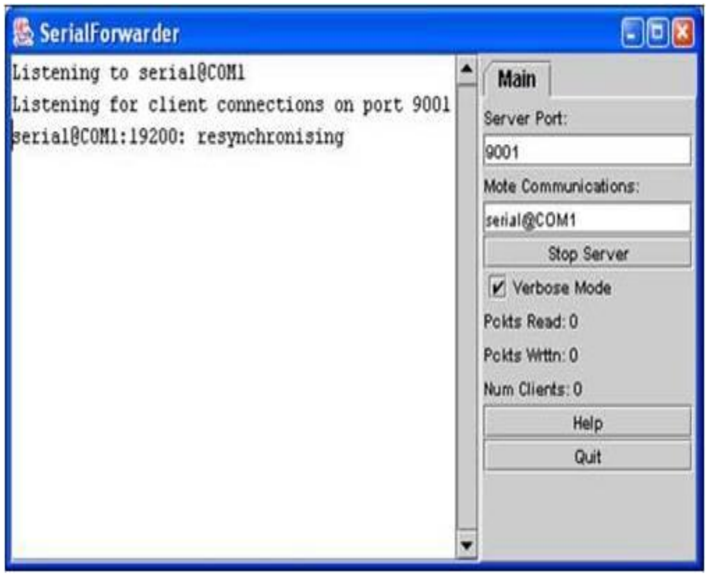

Figure 7: Serial Forwarder GUI

3.5.4 Xlisten

Xlisten is a user interface that serves as a tool to test the functionality of available data acquisition boards (DAQs). It displays the DAQ's output in a Cygwin window. Xlisten needs a DAQ board application and a driver, which can be modified to develop custom applications and to meet data logging needs. Xlisten is a program included with TinyOS installation. It can be found in the following folder: *c:\tinyos\cygwin\opt\tinyos-1.x\contrib\xbow\tools\src\xlisten.*

The boards drivers are located at *opt/tinyos-1.x/contrib/xbow/tos/sensorboards* and the XSensor Applications are located at: opt*/tinyos-1.x/contrib/xbow/apps/*. Xlisten can be used to test motes:

- Over the UART
- Over the RF

For a single mote configuration, the mote must be programmed with XSensorMTS310 application and plugged into the MIB510. The mote will stream packets over the UART or the radio. For the network of motes configuration, a base station mote needs to be programmed with TOSBase and plugged into the MIB510. All other motes need to be installed with an XSensorMTS310 application and put within range of the base station or a valid multihop peer. Xlisten must then be run with the -w flag to properly parse the wireless packets. Please refer to manual [16] for instructions on how to program a mote, and for a more complete discussion of the Xlisten display options. Proper care is to be taken to ensure that all the motes are programmed to the same frequency and group id.

3.6 Sensor data monitoring using Mote-View

During the installation of *Mote-View* a static database was included to make it possible to demonstrate *Mote-View's* features without having to be connected to an active sensor network or to be connected to a remote server/database [15]. *Mote-View* uses *XServe*™ data logging software to insert readings consolidated at a MIB510 interface gateway into the *PostgreSQL* database. A table in the database corresponding to the motes is selected. The sensor data can then be displayed within *Mote-View* in the form of a spreadsheet, chart, and topological map. The log to database option selected when connecting to a MIB510 enables us to view live sensor network data.

In order to have Mote-*View* display data from an active sensor network, the users must check on the "Live" checkbox. In Live mode, *Mote-View* refreshes the node list, charts, and topology views at a regular interval, the default being 10 seconds. *Mote-View*'s visualization tabs provide three ways to view the sensor data. The main display of the user interface consists of the four tabs:

- Data: Displays the latest sensor readings received for each node in the network.
- Command: The command tab provides the user with an ability to change different node parameters wirelessly.
- Chart View: The chart view provides the ability to generate graphs of a sensor reading against time for some set of nodes, with *y* axis being time and *x* axis being data in engineering units for the sensor readings.

- Topology View: The topology view shows a map of the network of motes, including placement and parenting information. This allows the user to define and view a topology of their mote deployment.

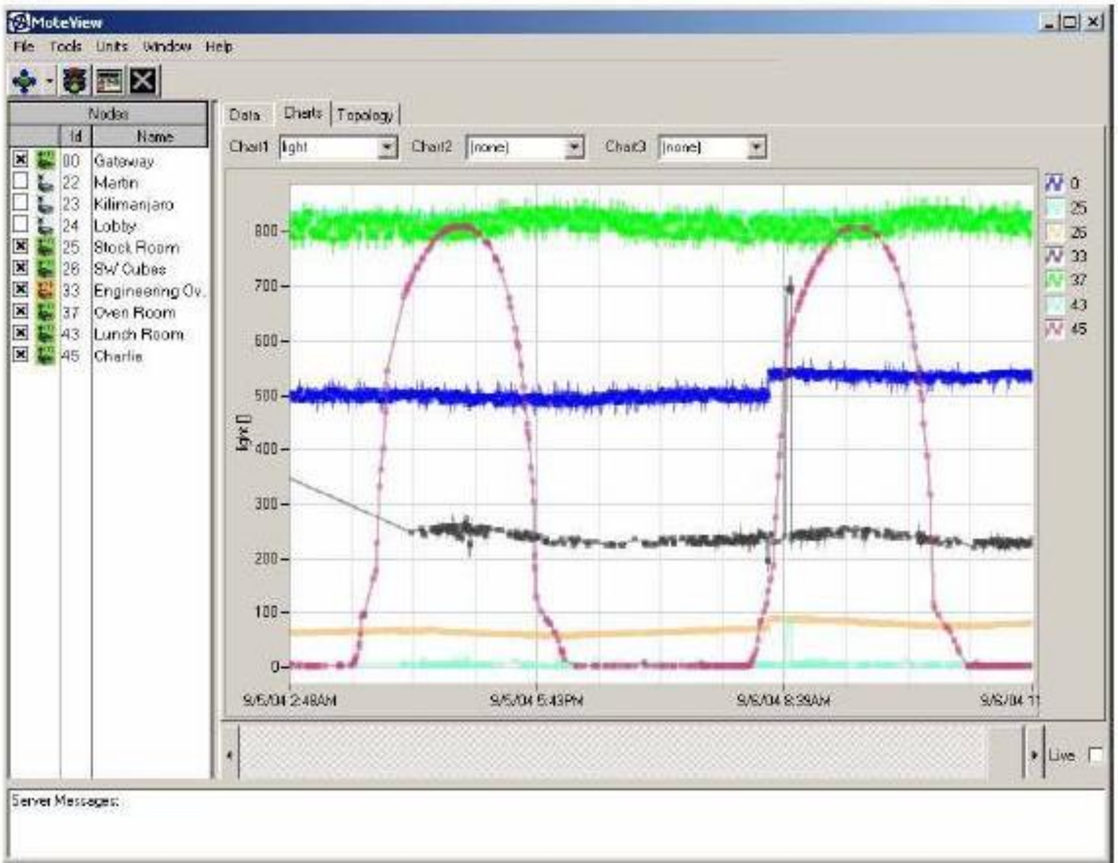

Figure 8: *Mote-View* window illustrating the charts feature

Always visible on the left hand side is the node list, which quickly shows the nodes that have been reported. Health of the network can be evaluated quickly by looking at the node list on the left. The colors of the node icon signify how long it has been since data has been received from that node.

Green - a healthy node reported within the last 20 minutes.

Red – inactive since 24 hours,

Orange – inactive in the past one hour

Yellow – inactive for the last 40 minutes

Moss green – inactive for the last 20 minutes

Gray - a mote that was added by the user, but has never reported in.

3.7 Sensor data retrieval

3.7.1 Using PostgreSQL and SQL commands

PostgreSQL is an advanced relational database system that is provided with the *Cygwin* on the PC and is available on the Stargate. The database tables that *Mote-View* accesses can be manipulated directly by advanced users. To access the *PostgreSQL* database, from a *Cygwin* shell type [15]

psql –h localhost –U tele task

Here ‘task’ is the name of the database

Once the database is accessed, the tables can easily be accessed using SQL commands. SQL is the generic command language used to manipulate databases such as *PostgreSQL*. SQL commands can be typed in directly from the *PostgreSQL* command shell.

PostgreSQL comes with other tools for offline manipulation of data besides the psql shell. The more useful of these are described here.

PostgreSQL Export: To output entire task database to a file, e.g., my_database.out

pg_dump –h localhost –U tele –f my_database.out task

Data can be exported in two different formats: XML or CSV (comma separated values).

To save contents of surge_results table to a file of SQL commands named surge.out:

pg_dump –h localhost –U tele –t surge_results –f surge.out task

PostgreSQL Import: To load files from a PostgreSQL exported table, use the following command:

psql task < surge.out

The output file can be read by a text editor, word processor, or spreadsheet program.

3.7.2 Using Xlisten and Matlab

Xlisten is the application that runs on the PC to obtain the data sent by the MIB510 board to the serial port. Xlisten is located at "/opt/tinyos-1.x/contrib./tools/src/xlisten/". To run Xlisten, change to the above directory where Xlisten is located. Compile Xlisten if necessary by entering the command "make". The file "xlisten.exe" shall appear if the program is compiled without any error. Enter the command

$ ./xlisten –p –s=com# >output_file.csv

The file "output_file.csv" would be saved to Xlisten directory as a comma separated variable file. The option **-s=com#** means Xlisten is using port #4. The port differs from computer to computer if a real com (serial) port is not used. Cooked option is used, **-c,** **t**he data from the sensors is converted to engineering units, which can be used for further analysis. Detailed information about the different modes and display options of Xlisten can be found in the manual [16].

The file saved using the redirection operator >, can be opened with excel to manipulate and graph the obtained data. If Xlisten is run again and the same name is given to the log file, the file 'output_file.csv' is overwritten by the new data values by Xlisten. The '*.csv'* files can be imported into Matlab for further data processing.

3.8 Assumptions and limitations of data collection using wireless sensors

During the course of the experiment, we have made certain assumptions that will be helpful in the sensor data processing.

- The experiment is carried in a dark room, so that any change in the conditions in the room (such as light and sound) could be attributed to that of the vehicle being monitored which we refer to as the *'target'.*

- The speed of the target is assumed to be fairly constant and it moves in a straight line. However, since the motion of the vehicle is controlled by a remote controller, it depends heavily on the operation of the remote controller with timely acceleration and deceleration.

- The time stamp at which the sensor data is logged into the database, the time registered in the video camera should coincide with the actual time.
- When a sensor reading exceeds the base value by a predetermined threshold value, in our case the threshold value is 20 ADC, a target is declared present.

3.9 Sensor data processing for vehicle detection

The testbed setup for this experiment involves both wireless sensors as well as the video cameras. A remotely controlled vehicle with two sensors and a flash light onboard will be monitored by Micaz motes. The coverage area is crossed by the vehicle assumed to be moving at constant velocity. We assume that only one vehicle is in the region during the time of experiment. The task of the testbed is to detect the vehicle and determine its motion trajectory. Flash light is pointed at motes equipped with light sensors that detect the presence of a passing vehicle. The Micaz motes with acoustic and light sensor functionalities are used as low resolution sensors deployed into the experimental area. The sensor measurements are analyzed to detect the presence of the vehicle. The sensor measurements are retrieved from the database using the methods described in Section 3.7.

The initial data collected from the environment (without the vehicle's presence) is used to set the base value (representing the condition of no vehicle presence). Later when the vehicle comes in the vicinity of the sensor the current sensor measurement is compared with the base value. The difference between the measurements is used to make the decision of the vehicle's presence. If the difference between both the measurements is greater than a preset threshold value, we declare the vehicle's presence. If the difference is less than the threshold, we declare that there is no vehicle present at that time instant.

When the target enters the coverage area, it could be detected by multiple sensor nodes depending on their distances from the target. Detection times are used to estimate the vehicle location and trajectory. As the vehicle moves through the coverage area, it interrupts the sensor’s field. The sensors detect the interruption times. If the vehicle is located closer to one sensor than the other, such sensor will measure a higher intensity. If the two sensors compare their measurements, they can determine the moment at which the object crosses in between them. Collecting such crossing times from different objects and sensor pairs would provide data that could be used to estimate the trajectories of the objects. If the sensor reading exceeds a threshold value chosen as 20 in ADC readings in the scenario, then we declare the vehicle's presence. Unfortunately, the sensor readings are not always accurate; this accounts to the possibility of missed detection and false alarms. In our experiment, we did observe that there were cases of missed detection and false alarms.

We used the set of light sensor readings from our experiments to determine the relationship of the sensor reading with the distance of light source, i.e. the moving vehicle. This relationship would later be used to estimate the target distance from sensor readings. The light sensor used in Micaz motes is sensitive to the angle at which light rays are incident on the sensor. This makes estimating distance from sensor readings hard, as the angle influences the reading. We observed variations in sensor readings in different directions even when the flash light was placed at an elevation (and hence angle) and the distance of the light source were kept constant. In order to solve this problem, we have mounted the flash light on the vehicle in such a way that it is at the level of the light sensor on the motes, ensuring the light is incident directly on the photo sensor.

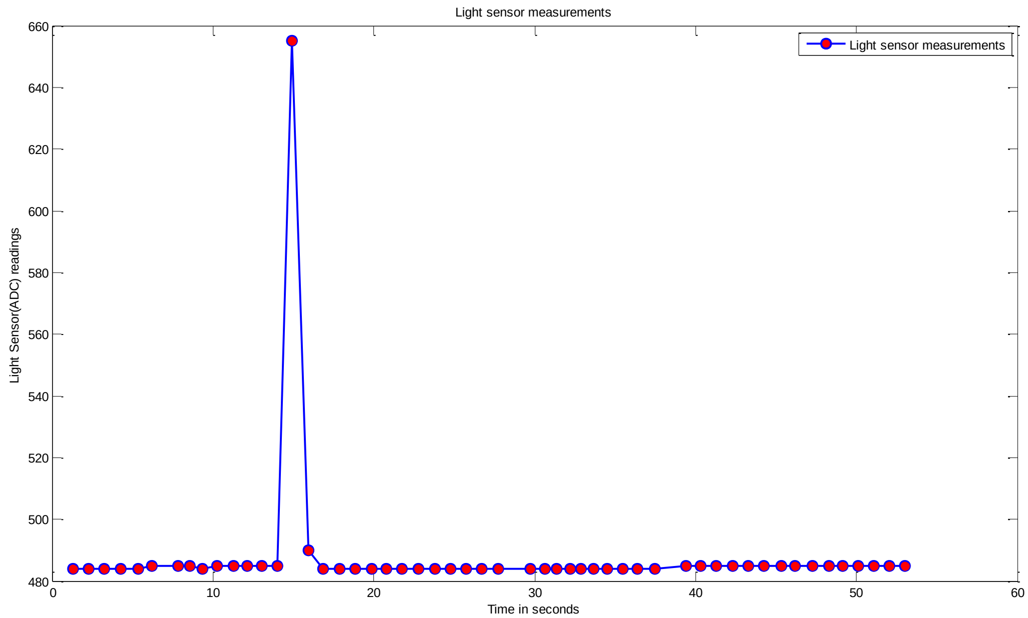

Figure 9: Light sensor measurements

Figure 9 shows the measurements of one particular light sensor placed along the coverage area during the experiment. We can observe that there is one peak over the threshold, which clearly indicates the vehicle's presence in the range of the sensor. However in some cases, the peak is either not large enough or is comparable with the noise level.

Each Micaz mote is attached with an MTS310 sensor board, of which the microphone is used for acoustic signal detection. The sensitivity of the microphone is rather poor; in the sense that a low-power sound source must be fairly close to the sensor, otherwise the microphone cannot distinguish the sound from ambient noise. Even with a nearby source, the chance of detection by the device is probabilistic at best. Tone detectors in Micaz are extremely unreliable; they sometimes miss the presence of a sound signal emitting from a nearby source. The tone detector on the Mica sensor board detects a specific range of sound frequencies and provides binary output indicating the presence or absence of a signal.

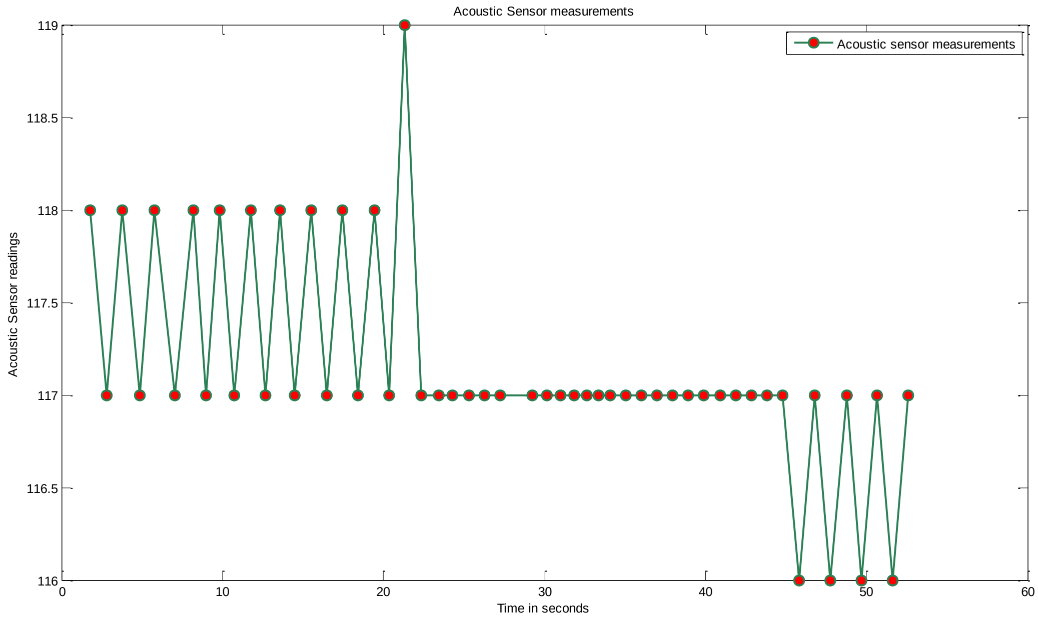

Figure 10: Acoustic sensor measurements

Figure 10 shows the measurements of a particular acoustic sensor placed along the coverage area during the experiment. We can observe that there is no significant change in the measurements which indicates the vehicle's presence in the range of the sensor. The difference between the peak and the other measurements is atleast not as large as the threshold value; it is more comparable with the noise level.

In our experiments, we also mounted two Micaz motes onboard the moving vehicle. The 2-axis accelerometer is used for movement measurement, i.e. to gain knowledge of the true accelerations invoked by the remote controller, which is helpful to create the ground truth. The raw accelerometer readings did not help in providing useful information about the vehicle's motion. Magnetic fields can be measured using the 2-axis magnetometer which is sensitive enough to measure the earth's magnetic field. In our experiment, the magnetometer readings did not indicate the presence of magnetic field in the vicinity for majority of the sensors.

3.10 Observations

The following observations are made from the experiment and the sensor data obtained.

- We found that the light sensor used in Micaz motes is sensitive to the angle at which light rays are incident on the sensor. To reduce this sensitivity, we experimented with the light source mounted on the vehicle in such a way that it is at the level of the light sensor on the motes. The light source was always kept at the same height.

- From the sensor measurements obtained during the experiment, we can say that the light sensor accuracy is approximately in the range of 1.2ft (0.36m). This value is estimated using the sensor measurements and the physical location, distance between sensors in the experimental setup. Beyond this distance the sensor reading is very low to be reliable. The vehicle must be closer than this distance for a sensor to be able to detect it.

- To ensure that the light source always emits light with the same power, we used freshly recharged batteries for all experiments. The battery in the motes gets discharged very quickly. We have observed many times during our experiments, that some of the sensors had no readings logged to the database due to the low battery condition. When we replaced the old batteries with freshly charged batteries, the sensors worked just fine for the next round of the experiment.

- Inorder to check the sensitivity of the sensors, we placed the flashlight on the vehicle facing the same direction in both onward and return path. Since we have the ten sensors placed equally on both sides of the coverage area, only the sensors which faced the light directly, were able to sense changes in the light conditions, while the sensors on the other side of the coverage area were not able to sense any changes in the external conditions in the room since the flashlight was still pointing towards the other direction

- In another scenario, when the direction of the flashlight is changed in the onward and return path, the sensors on respective sides were able to sense the changes in the light conditions. We repeated the scenarios number of times to test the sensitivity and range of the sensors.

- Since the motion of the vehicle is controlled by a remote controller; it depends heavily on the operation of the remote controller with timely acceleration and deceleration. The vehicle was moving approximately at a rate of 1m/s. With little increase in the acceleration, the vehicle moves little faster approximately at a rate of 3 m/s. Some of the sensors completely missed detecting the changes in the lighting and acoustic conditions in the coverage area caused by the presence of the vehicle.

- The sensitivity of the microphone was rather poor; in the sense that a low-power sound source must be fairly close to the sensor, otherwise the microphone cannot distinguish the sound from ambient noise. Even with a nearby source, the chance of detection by the device is very uncertain. Hence we only considered light sensors for vehicle detection.

- The localization was done by assuming that the correct decision was made always, i.e., the locations of the vehicle were estimated based on the known sensor locations once the target was declared present.

- There is a noticeable time delay between when the sensor readings were logged into the database and the actual passing by of the vehicle. This time delay is on the order of 0.36sec when averaged over 6 particular instants of the vehicle passing by the sensors. We compared the timestamp in the sensor readings to the timestamp in the video cameras for the corresponding instants.

At instants when the sensor reading is significantly higher (difference between the current sensor reading and the normal base readings is greater than the predetermined threshold value), the target is declared to be present or detected. The localization is done by assuming that the locations of the vehicle can be estimated based on the known sensor locations once the target is declared present. Light readings are used to make the decision when the vehicle is present.

Chapter 4 –Video based Vehicle Tracking

4.1 Background Information

Video surveillance applications call for the real-time observation of objects (humans or vehicles) in a given environment (indoor, outdoor, or aerial). The continuous observation of the object leads to a description of its activities within the environment. A complete video surveillance system typically consists of foreground segmentation, object detection, object tracking, object analysis, recognition and activity analysis [17]. For any video surveillance problem, effective segmentation of foreground (region of interest in the image) and background plays a key role in subsequent detection and tracking results. A traffic surveillance system needs to detect and track the vehicles and also classify them if needed. The processing of a video stream for characterizing events of interest relies on the detection, in each frame, of the objects involved, and the integration of this frame based information to model simple and complex behaviors.

In the computer vision literature, the different tracking approaches using video sequences can be broadly classified under the following categories. A more detailed explanation about the different camera based tracking approaches can be found in [18]:

Region based tracking: A background model is generated before the start of the experiment. For each input image frame, the absolute difference between the input image and the background image is processed to extract foreground region corresponding to the objects or "*targets*" to be tracked. The idea here is to identify a connected region in the image, "a blob" associated with the *target* and then track it over time using a cross-correlation measure. Initialization of the process is most easily done by the background subtraction technique. This method is also known as blob tracking.

Active contour based tracking: A dual to the region based approach is tracking based on active contour models, or snakes representing the boundary of an object. The idea is to have a representation of the bounding contour of the object and keep dynamically updating it. These algorithms provide efficient descriptions of objects compared to blob tracking. However, these algorithms have drawbacks, for example, they do not work well in the presence of occlusion and their tracking precision is limited by a lack of precision in the location of the contour. If one could initialize a separate contour for each object, then one could track even in the presence of partial occlusion.

3D model based tracking: Model-based tracking algorithms localize and recognize objects by matching a projected model to the image data. For visual surveillance in traffic scenes, 3-D model-based vehicle-tracking algorithms have been studied widely and 3-D wire-frame vehicle models were adopted. In [19], a single vehicle is successfully tracked through a partial occlusion, but its applicability to congested traffic scenes has not been demonstrated. The main advantages of the algorithms based on 3-D models are that they are robust even under interference between nearby image motions, they naturally acquire the 3-D pose of vehicles under the ground plane constraint and hence can be applied in cases in which vehicles greatly change their orientations. However, they also have some disadvantages, such as the requirement for 3-D models, high computational cost.

Feature based tracking: Feature-based tracking algorithms perform the recognition and tracking of objects by extracting elements, clustering them into higher level features, and then matching the features between images. The advantage of this approach is that even in the presence of partial occlusion, some of the sub-features of the moving object remain visible. The system proposed in [18] automatically detects and tracks feature points throughout the image sequence, estimates the 3-D world coordinates of the points on the vehicles, and groups those points together in order to segment and track the individual vehicles. Experimental results shown there demonstrated the ability of the system to segment and track vehicles in the presence of occlusion and perspective changes. Overall, these algorithms claim to have a low computational cost compared to other tracking algorithms. However, they too have a number of drawbacks. The recognition rate of vehicles using two-dimensional image features is low, because of the nonlinear distortion due to perspective projection, and the image suffers from variations due to movement relative to the camera. Also, they generally are unable to recover the 3-D pose of vehicles.

There is interest in vehicle tracking due to its applications for public safety and traffic monitoring and surveillance in general. Currently deployed cameras in such applications can be controlled remotely with the ability to pan and zoom around a given location, allowing monitoring of a greater area, compared to other monitoring techniques. 3-D tracking aims at continuously recovering all six degrees of freedom that define the camera position and orientation relative to the scene, or, equivalently, the 3-D displacement of an object relative to the camera. In the following sections, we will describe the tracking of a target using multiple uncalibrated cameras.

4.2 Camera model

Recovering 3-D structure from images becomes an easier problem when the images are taken with *calibrated* cameras. A camera is said to be *calibrated* if the mapping between image coordinates and directions relative to the camera center are known. However, the position of the camera in space (i.e. its translation and rotation with respect to world coordinates) is not necessarily known. For an ideal pinhole camera delivering a true perspective image, this mapping can be characterized completely by the intrinsic parameters of the camera. Camera's extrinsic parameters represent its location and rotation in space. The intrinsic camera parameters are:

- The x-coordinate of the center of projection, in pixels (Ox)
- The y-coordinate of the center of projection, in pixels (Oy)
- Pixel width and height (*fx*, *fy*).
- Angle between the optical axes (s)

A camera is usually described using the pinhole model [20]. Mathematically, image formation can be defined as projection from 3-D space to image plane. There exists a collineation which maps the projective space to the camera's retinal plane: $P^3 \rightarrow P^2$.

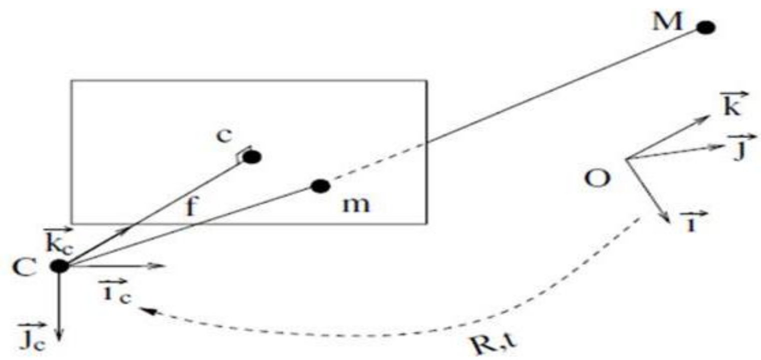

Figure 11: Camera perspective model

- Here $(O,\vec{i},\vec{j},\vec{k})$ is in the world coordinate system and $(C,\vec{i}_c,\vec{j}_c,\vec{k}_c)$ in the camera coordinate system.
- The coordinates of a 3-D point M = $[X\ Y\ Z]^T$ expressed in a euclidean world coordinate system and the corresponding 2-D point m = $[u\ v]^T$ in the image are related by

$$s\tilde{m} = P\tilde{M} \qquad \text{(Equation 1)}$$

- Here s is a scale factor, $\tilde{m}$ = $[u\ v\ 1]^T$ and $\tilde{M}$ = $[X\ Y\ Z\ 1]^T$ are the homogeneous coordinates of points m and M, and P is a 3 × 4 projection matrix.
- P is usually taken to be a perspective projection matrix because it realistically describes the behavior of a reasonably good quality camera.
- The optical axis passes through the centre of projection (camera) *C* and is orthogonal to the retinal plane. The point ***c*** is called the principal point, which is the intersection of the optical axis with the retinal plane. The focal length *f* of the camera is also shown, which is the distance between the centre of projection and the retinal plane.
- If only the perspective projection matrix ***P*** is available, it is possible to recover the coordinates of the optical centre or camera.

- The world coordinate system is usually defined as follows: the positive Y direction is pointing upwards, the positive X direction is pointing to the right and the positive Z direction is pointing into the page.

Using the above equations, we can represent the world coordinates in camera coordinates and vice versa. We assume that the camera rotation angles and the location of the camera are known to us. Once the camera external parameters are known, with the help of above equations and the known reference points in both 3-D world and 2-D pixel coordinates, the camera internal parameters can be calculated.

4.3 Overview of Camera calibration

From a mathematical point of view, an image is a projection of a three dimensional space onto a two dimensional space. Geometric camera calibration is the process of determining the 2-D to 3-D mapping between the image and the world coordinate system [21]. Therefore, obtaining the 3-D structure of a scene depends critically on having an accurate camera model. In the case of a simple pin-hole camera, 6 extrinsic (the position and orientation of camera in some world coordinate system) and 5 intrinsic parameters (principal point, angle between the axes and pixel width and height) describe fully camera calibration.

Camera calibration is an important part of most computer vision systems. We can classify those techniques roughly into two categories: photogrammetric calibration and self-calibration. In photogrammetric calibration approaches, calibration is performed by observing a calibration object whose geometry in 3-D space is known with very good precision. Calibration can be done very efficiently using this approach. Self-calibration techniques do not use any calibration object. Although the equations governing the transformation from 3-D world coordinates to 2-D image coordinates are nonlinear functions of intrinsic and extrinsic camera parameters, they are linear if lens distortion is ignored. This made it possible to compute a perspective transformation matrix first using linear equations.

The majority of the self-calibration algorithms [22] described in the CV literature treats intrinsic camera parameters as constant but unknown. The intrinsic parameters can be extracted from the images themselves. Camera's intrinsic parameters (e.g. focal length, pixel width and height and the optical center) and its position in the world coordinate system are not known in advance. All these parameters are to be determined using the equations governing the transformations from 3-D world coordinates to 2-D pixel coordinates. Using these parameters, ground plane rectification can be done. If a pixel in the image appears on the ground plane, its 3-D coordinates can be found in the world reference frame.

4.3.1 Camera calibration matrix / Internal parameters matrix

- The '**K'** camera calibration matrix contains the intrinsic camera parameters**,** also called internal parameters. This matrix is used to convert between the retinal plane and the actual image plane. We write

$$\mathbf{K}=\begin{bmatrix} f_x & \mathbf{s} & O_x \\ \mathbf{0} & f_y & O_y \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \qquad \text{(Equation 2)}$$

- fx and fy are respectively the scale factor in the x and y coordinate directions in pixels. They are proportional to the focal length f of the camera and are the width and height of pixels in the image.

- **c** = $[Ox, Oy]^T$ represents the image coordinates of the intersection of the optical axis and the image plane, also called the principal point;

- ***s***, referred to as the skew, is non-zero only if the x and y directions are not perpendicular, which is rare in modern cameras.

4.3.2 External parameters matrix`

- The 3×4 external parameters [**R** | **t**] matrix defines the orientation and the position of the camera. It is formed of **R** rotation matrix and **t** translation vector and is often referred to as the *camera pose*.

- It corresponds to the euclidean transformation from a world coordinate system to the camera coordinate system.

- A 3-D point represented by the vector $\mathbf{P}_w$ in the world coordinate system will be represented by the vector $\mathbf{P}_c = \mathbf{R}\mathbf{P}_w + \mathbf{t}$ in the camera coordinate system.

- From this relation, we can easily recover the expression of the c*amera center*, or *optical center* **C** in the world coordinate system. It must satisfy $\mathbf{0} = \mathbf{RC} + \mathbf{t}$, which implies $\mathbf{C} = -\mathbf{R}^{-1}\mathbf{t} = -\mathbf{R}^{T}\,\mathbf{t}$.

4.4 Different methods of estimating camera parameters

Different techniques to determine the internal and external camera parameters are discussed. In most 3-D tracking methods, the internal parameters are assumed to be fixed and known, which means that the camera cannot zoom, because it is difficult to distinguish a change in focal length from a translation along the camera *Z*-axis. These parameters can be estimated during an offline camera calibration stage, from the images themselves.

Classical calibration methods make use of a calibration pattern of known size inside the field of view. Sometimes it is a 3-D calibration grid on which regular patterns are painted [23]. The 3-D coordinates of the corners of the white squares with respect to the grid's upper left corner are exactly known. It is relatively easy to find those corners in the image and, from the correspondence between the 3-D points and the 2-D image points, to compute projection matrix parameters. One image and at least six non-coplanar feature 3-D points, manually selected or automatically detected on the acquired image, are needed. Each correspondence between a 3-D scene and 2-D image point provides one equation. The solution solves an over-determined system of linear equations.

In [24] an algorithm is described which requires at least two different views of a planar pattern. An even more accurate calibration is obtained using a large number of views (twenty or more). If images are taken by the same camera with fixed internal parameters, correspondences between three images are sufficient to recover both the internal and external parameters which allow us to reconstruct 3-D structure up to a similarity [25]. In [26] a camera calibration approach using vanishing lines is presented.

Self-calibration techniques do not use any calibration object. A set of points whose coordinates in the three dimensional World coordinate systems and whose pixel coordinates in the two dimensional image plane are known are used to estimate the camera parameters. The determination of camera external parameters, i.e. the location and orientation means to relate the camera-centered coordinate system to the world coordinate system by rotations followed by a translation. The intrinsic camera parameters relate the pixel coordinates in the image plane to the corresponding points in the camera coordinate system. The main advantage of these methods is that they eliminate nonlinear optimization. However, lens distortion cannot be modeled using these methods, and the number of unknowns in linear equations is generally much larger than the actual degrees of freedom.

4.5 Preprocessing techniques

Preprocessing of the video sequence is necessary to improve the detection of moving objects, filtering out most of the unwanted motion in the video. Performing a statistical procedure on the frames gives us a set of pixel coordinates representing the centroids of the moving vehicle, with respect to the three cameras. In this section, different preprocessing techniques performed are discussed.

4.5.1 Processing of video files in Matlab

The output of the digital camera is an '*.mpeg*' video file. To read '*.mpeg'* video files in Matlab '*.mpeg*' files are converted to '*.avi*' files using a Movie Converter [27] (it converts '*.mpeg'* files to '*.avi'* files). Avi files can be easily read into Matlab using the following commands: *aviread*, *aviinfo*. After the conversion is done, the *'.avi'* file is loaded into Matlab. The command aviread reads the AVI movie into the MATLAB movie structure, Mov. Mov has two fields, "*cdata*" and "*colormap*". *Aviinfo* is a very useful command, which returns a structure whose fields contain information about the *avi* file. Following is the information that can be retrieved from *aviinfo* command.

- Filename, FileSize, FileModDate
- NumFrames, FramesPerSecond
- Width, Height

- ImageType, VideoCompression
- Higher quality numbers indicate higher video quality, where lower quality numbers indicate lower video quality.
- NumColormapEntries.

We can also read only the frame(s) specified by a particular *index*. It can be a single index or an array of indices into the video stream, where the first frame is number one. The video sequences were converted to set of images at a rate of 30 fps.

4.5.2 Vehicle extraction from images

The initial stage of the surveillance problem is the extraction of moving targets from a video stream. The high level description of a video stream relies on accurate detection and tracking of the moving objects, and on the relationship of their trajectories to the scene. Background subtraction is the process of separating out foreground objects from the background in a sequence of video frames. Background subtraction is used in many emerging video applications, such as video surveillance and traffic monitoring. Many methods exist for background subtraction, each with different strengths and weaknesses in terms of performance and computational requirements.

There are three conventional approaches to automated moving target detection [28]: temporal differencing (two-frame or three-frame), background subtraction; and optical flow. Temporal differencing is very adaptive to dynamic environments, but generally does a poor job of extracting all relevant feature pixels. Background subtraction provides the most complete feature data, but is extremely sensitive to dynamic scene changes due to lighting and extraneous events. Optical flow can be used to detect independently moving targets in the presence of camera motion; however, most optical flow computation methods are very complex and are inapplicable to real-time algorithms without specialized hardware.

We have used the low-complex background subtraction method. Since our experiment was conducted in a closed room, there were not many changes in the external environmental conditions such as lighting, other objects appearing into and away from the scene. And the flash light mounted on the moving vehicle was the only known source of light in the closed room. This method subtracts out extraneous background noise between the frames. The background frame can be obtained offline from the images or the video before the start of the experiment.

4.5.3 Calculation of vehicle centroid

Background subtraction procedure gives the foreground pixels. We further need to cluster the foreground pixels into the vehicle. We initially use morphological operations of image opening. Image opening operation removes potential noise foreground pixels that connect two clusters. After these operations we remove the foreground clusters having an area less than 10 pixels and use connected component labeling which gives us the potential vehicle in the scene. The procedure is explained in detail below.

The absolute difference between the two frames can be used to divide an image frame into changed and unchanged regions. Since only the vehicle moves, we expect the changed region to be associated only with the vehicle. To begin, we convert each frame to grayscale using *rgb2gray*.

If f (:,:, k) is the original frame, g(:,:,k) is the frame in grayscale at the k^{th} instant.

$$g(:,:,k) = rgb2gray(f(:,:,k)); \qquad \text{(Equation 3)}$$

Next, we compute frame differences using *imabsdiff*.

$$d(:,:,k) = imabsdiff(g(:,:,k), f(:,:,k)); \qquad \text{(Equation 4)}$$

Here d(:,:,k) is the difference between the two frames at the k^{th} instant.

The function *graythresh* computes a threshold that divides an image into background and foreground pixels. Since *graythresh* returns a normalized value in the range [0,1], we must scale it to fit our data range, [0,255] to eliminate any of the shadow spots in the figure.

$$thresh = graythresh(d) \qquad \text{(Equation 5)}$$

$$\begin{aligned} &bw = (d >= thresh * 255); \\ &imview(bw(:,:,1)); \end{aligned} \qquad \text{(Equation 6)}$$

If the resulting binary image has small extra spots, these can be removed using the following technique. The technique we used, area opening, removes objects in a binary image that are too small. The call to *bwareaopen* removes all objects containing fewer than 10 pixels.

$$bw2 = bwareaopen(bw, 10, 8); \qquad \text{(Equation 7)}$$

The third argument, 8, tells *bwareaopen* to assume that pixels are connected only to their immediate 8 neighbors in each frame. *bwareaopen* will then treat *'bw'* as a sequence of two-dimensional images rather than one three-dimensional image. Finally, we label each individual object (using *bwlabel*) and compute its corresponding center of mass (using *regionprops*).

$$s = regionpros(bwlabel((bw2(:,:,1)),'centroid'); \qquad \text{(Equation 8)}$$

$$c = [s.Centroid]; \qquad \text{(Equation 9)}$$

regionprops measures properties of image regions (blob analysis)

The variable *c* is a structure array containing the desired measurements for each object satisfying the criteria specified in the above procedure, i.e objects containing greater than 10 pixels. The resultant array *c* contains more than one measurement corresponding to each of the objects present in the image. Only one object, the vehicle is of prime importance to us. Hence we can select only the vehicle region based on certain criteria such as select regions whose area is greater than say *'x'*.

The function ismember is useful in conjunction with regionprops for this purpose.

$$idx = find([stats.Area] > x); BW2 = ismember(L, idx); \qquad \text{(Equation 10)}$$

Using the above *procedure*, only the centroid of the vehicle is calculated in the last picture in Figure 12. The other small white spots that can be seen are ignored in the process of centroid calculation. The objects other than the moving vehicle are eliminated in the centroid calculation procedure by accessing the object properties such as *Centroid*, *FilledArea* and *Area*. Some examples of such objects are the difference between timestamps of the current and background frames and the light from the flashlight. Since the object related to the difference in timestamps occurs more or less at the same set of pixels in the frame, the centroid of the object in every frame lies within a range. We used this range of the centroid to remove the object during the vehicle centroid calculation.

Figure 12: Extraction of an object from the background

In the figure 12, the estimated background image is subtracted from the image frame being processed, to get outline of the moving object. The background frame is obtained offline from the video before the start of the experiment. A statistical operation on the current frame returns the centroid of the object that is the outline of the moving vehicle.

The frame rate for the set of images obtained from the video cameras is 30fps. Since the vehicle is moving slowly at a constant velocity, in order to have a continuous trajectory of the moving vehicle, the vehicle is extracted from the images using the frame differencing method once in every 30 frames. The total duration of the experiment was about 54 seconds. Some of the set of frames were not considered, as they represent the manual adjustments of the vehicle, etc, during the experiment. The accuracy of the vehicle centroid calculation method is affected by the presence of any other sources of light in the room during the experiment. Changes in the light conditions in the room would lead to different objects in addition to the vehicle to be detected using the frame differencing method. In order to avoid detecting non-stationary background objects such as moving leaves, shadows, in future we would need to have more robust background modeling blob extraction techniques.

A set of pixel coordinates representing the centroid of the vehicle is obtained for each of the camera. The trajectory formed by these pixel coordinates can be used to estimate the trajectory describing the vehicle's motion.

Chapter 5 –Experimental Study

5.1 Experimental setup

The wireless sensors and the video cameras are used to detect and track the moving vehicle. The wireless sensors are deployed in such a way that they complement the video cameras with additional information about the moving vehicle. The three cameras are placed in such a way that the vehicle is in the field of view of atleast one camera all the time. The three cameras have overlapping field of view, providing redundant information about the vehicle and the coverage area. The information generated from overlapping views can be used to minimize the ambiguities of occlusion as well as to improve the accuracy of the position estimate of the vehicle. The moving vehicle has a flashlight and two wireless sensors onboard. The video cameras are mounted on fixed locations, recording video streams and directly communicating to a desktop. The laptop enables the processing of the sensor programming board collecting sensor data from each Micaz mote. The sensor data is processed to enable vehicle detection and the video sequences generated by the cameras are processed to obtain a set of images. The moving vehicle is detected in each of the images using the background subtraction method. The centroid of the vehicle is calculated which represents the vehicle location in the 2-D image plane. In order to generate the vehicle trajectory in the real world plane, the 2-D pixel coordinates corresponding to the vehicle centroid are mapped to the vehicle locations in the real world.

5.2 2-D to 3-D mapping

The trajectories obtained by video cameras represent the motion of the vehicle. The objective is to detect the presence of the vehicle and find the locations at different time instants by using the measurements from wireless sensors. The 2-D observations from each camera view will be used to form global 3-D world coordinate object tracks. Once the 2-D observations have been merged it is then possible to track each object in explicit 3-D coordinates. An algorithm to obtain the 2-D pixel coordinates from the video sequences and to map them to the corresponding 3-D world coordinates is presented below. This procedure is followed individually with respect to the three cameras.

We assume that we have a set of points whose coordinates in the 3-D world coordinates (XYZ) and the 2-D image coordinates (*UV*) are known. In addition to these points, we also know that the view is a perspective projection, and therefore the transformation of the three-dimensional coordinates of a point to the two-dimensional coordinates can be expressed compactly as a 3 x 4 transformation matrix using homogeneous coordinates.

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = w' \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \text{(Equation 11)}$$

Notice that there are 12 elements in the transformation matrix, and that the measurements from the images are not the values, x’, y’ and w’ as shown in but the ratio, U = x' / w' and V = y' / w'. Infact we have no way of knowing the value of w', and different input points, X Y Z may give different values of w'.

The transformation matrix can be expressed as a product of the camera internal and external parameters matrix. The location of the camera is assumed to be known and is fixed. The range of the camera rotation angles is assumed to be known; hence the camera rotation matrix is computed. Since the location and orientation of the camera are assumed to be known, the external parameters matrix is computed.

- Using the external parameters matrix and the world coordinates, the camera coordinates are computed: Camera coordinates = External $\times$ World coordinates.
- Using the camera coordinates and the pixel coordinates, camera internal parameters matrix are calculated: pixel coordinates = Internal $\times$ Camera coordinates.

5. 3 Self calibration procedure

As discussed previously, camera calibration is a very important part of many computer vision systems. Accuracy of the calibration dictates the accuracy of the overall system. It is equally important that the calibration process should be simple and need not require special calibration objects. There are situations where in the internal parameters and the camera pose are unknown. In such cases, having the calibration object solves the problem of calibrating intrinsic camera parameters. But we would still need to know information such as height of the camera from the ground plane and its orientation with respect to the ground plane. If the video taken from an unknown camera at an unknown location is to be processed offline, the camera parameters are to be determined from the video sequence itself. This requires that a self calibration approach needs to be performed.

Self-calibration refers to the process of calculating all the intrinsic parameters of the camera using only the information available in the images taken by that camera [22], [29]. No calibration frame or known object is needed: the only requirement is that there is a static object in the scene. Sometimes more than one view of the same picture is used to estimate calibration parameters (For example, Stereo [30]). Most camera parameters can be estimated from the measurements of a single image when sufficient geometric object knowledge is available.

Suppose that a camera observes "*n*" geometric features such as points or lines with known positions in some fixed world coordinate system. We will:

- Compute the perspective projection matrix P associated with the camera in this coordinate system.

- Compute the intrinsic and extrinsic parameters of the camera from this matrix.

In our experiment setup, we have three different cameras positioned at different locations, covering partially the scene for the duration of the experiment. From the video sequence obtained by each video camera, a set of pixel coordinates representing the centroids of the moving vehicle in the image is calculated. The transformation from the pixel coordinates to a point on the camera coordinate system can be expressed with the help of the camera's internal parameters. The camera's internal parameters link the pixel coordinates of an image point with its corresponding camera coordinates. The locations of the three cameras with respect to the real world coordinate are measured prior to the experiments. The cameras are located at 0ft, 15ft and 23ft respectively. The cameras are located at a height of 3ft approximately.

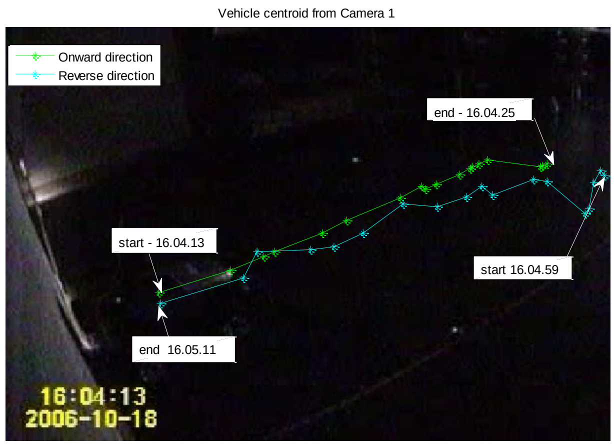

Figure 13: Plots of vehicle centroid from camera 1

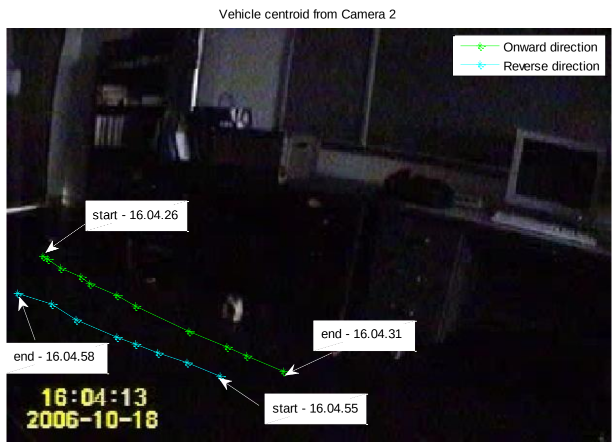

Figure 14: Plots of vehicle centroid from camera 2

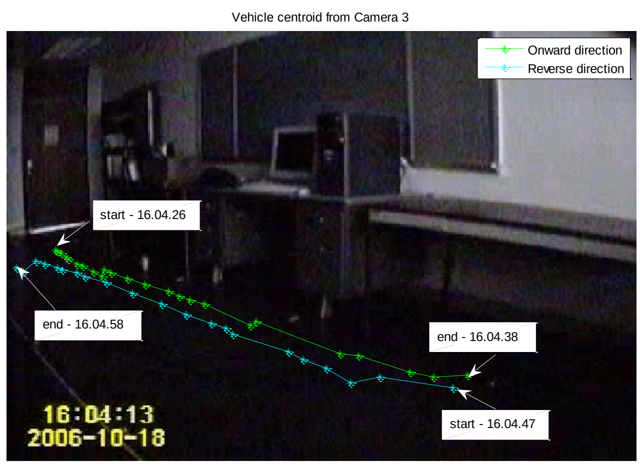

Figure 15: Plot of vehicle centroids from camera 3

Cameras 1 and 2 cover only a part of the coverage area. The vehicle is seen from this camera for few seconds in the onward and return directions respectively, as indicated in the figures. Camera 3 covers most of the vehicle's trajectory in the coverage area, except for a few seconds in the beginning. The above figures show the trajectory of the moving vehicle in both the onward and return paths as obtained using the vehicle centroid coordinates. The centroids obtained from the statistical filtering approach (based on area) are plotted. The trajectory is clearly marked up in the figure.

We use the 2-D observations from each camera view to obtain global 3-D world coordinate object tracks. In order to compute the calibration parameters of the three cameras, each camera is calibrated separately. The position of the reference point, "clock" is known in the real world coordinates, and the pixel coordinates of the reference point with respect to the three cameras is known. Using the world and pixel coordinates of the reference point and the perspective projection equations, the camera pixel width and height can be estimated.

Let Pw be a point in the 3-D world coordinate system, Pc be its corresponding point in the camera coordinate system. The transformation between the two coordinate systems can be represented by the following equation:

$$\mathbf{P}_c = \mathbf{R}\mathbf{P}_{\mathbf{w}} + \mathbf{T} = \begin{pmatrix} r_{11}X_w + r_{12}Y_w + r_{13}Z_w + T_x \\ r_{21}X_w + r_{22}Y_w + r_{23}Z_w + T_y \\ r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z \end{pmatrix} = \begin{bmatrix} \mathbf{R}_1^T\mathbf{P}_{\mathbf{w}} + T_x \\ \mathbf{R}_2^T\mathbf{P}_{\mathbf{w}} + T_y \\ \mathbf{R}_3^T\mathbf{P}_{\mathbf{w}} + T_z \end{bmatrix} \qquad \text{(Equation 12)}$$

Here, T is the translation vector in terms of X, Y, Z directions and R is the rotation matrix. The orientation of the camera is represented in a spherical space, where ψ**,** represents the camera optical-axis rotation, θ represents the angle between the optical axis of the camera and the Zw axis of the world coordinate W and φ represents the angle between the Xw axis and the projection of the camera optical axis onto the xz-plane of and the world coordinate. It is also known as the Azimuth angle. Here the camera optical axis rotation ψ is assumed to be zero. The Rotation matrix can be represented by:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi\cos\theta & \cos\varphi\cos\theta & \sin\theta \\ \sin\varphi\sin\theta & -\sin\theta\cos\varphi & \cos\theta \end{bmatrix} \qquad \text{(Equation 13)}$$

Azimuth is an angle in the xz plane with positive value indicating counter clockwise rotation of the viewpoint. Elevation is the angle above (postive) or below (negative) the x-z plane. In our experiment, we assumed that the vehicle is present in the xz plane. Since the cameras are pointing downwards in our case, the elevation angles are negative. The range for the angles is given as below:

azimuth - (0 to 360 degrees)

elevation (-90 to + 90 degrees)

The world coordinate system is usually defined as follows: the positive Y-direction is pointing upwards, the positive X-direction is pointing to the right and the positive Z-direction is pointing into the page. The camera location represents the physical location of the camera in the real world plane. This is known before the experiment. The transformation between the camera coordinates and the image coordinates can be represented by the following equation.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \qquad \text{(Equation 14)}$$

Camera internal parameters matrix, $K = \begin{bmatrix} f_x & s & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$ (Equation 15)

We assume that the optical axis of the camera passes through the center of the image. The location of the camera is known and is fixed. Since the three cameras used are of the same type, ideally the camera intrinsic parameters should be the same. The pixel width and pixel height are assumed to be the same for the three cameras. Substituting the known quantities in the algorithm, the unknown parameters are to be estimated. The pose of the three cameras is assumed to be known. We have a total of two unknowns, pixel width and pixel height of the set of cameras.

Here, *fx* and *fy* are the focal length of the camera expressed in terms of pixel width and pixel height respectively and (Ox,Oy) is the camera optical center. Once the Internal parameters are known, 2-D to 3-D mapping is done as follows:

$$x' = x_{im} - o_x = -f_x \frac{r_{11}X_w + r_{12}Y_w + r_{13}Z_w + T_x}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z}$$
$$y' = y_{im} - o_y = -f_y \frac{r_{21}X_w + r_{22}Y_w + r_{23}Z_w + T_y}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z} \qquad \text{(Equation 16)}$$

Using the knowledge of the world and pixel coordinates for the reference point, the internal parameters for the three cameras are calculated. Using the calibration results of each camera, the pixel coordinates are first converted to camera coordinates using the internal parameters. The camera coordinates are then converted to the corresponding world coordinates using the external parameters. The external parameters define the location and orientation of the camera coordinate system with respect to the world coordinate system.

The results of the Self calibration procedure with respect to the three cameras are presented in the table. The optical center for the three cameras was initially taken to be at the image center i.e., (176,120). In order to have the same pixel width and pixel height and also to be able to map the 3-D location of the reference point accurately into the 2-D image coordinates; the optical center has been moved appropriately.

Camera parameters	*fx* (pixelwidth) num of pixels	*Fy* (pixelheight) num of pixels	Camera optical center (u_0,v_0) pixels	Camera rotation angles	Camera Location w.r.t world frame
Camera 1	12.9	15.2	(170,90)	21,-46 degrees	(0,4ft,-1ft)
Camera 2	15.5	10.8	(100,170)	88,-66 degrees	(15ft,4ft,-1ft)
Camera 3	13.8	9.2	(40,135)	83,-64 degrees	(23ft,4ft,-1ft)

Table 1: Camera parameters - Self calibration procedure

In order to provide high-quality video sequences for chosen geographic objects in a coverage area, the vehicle speed should not be too fast. The output is a set of synchronized, time stamped video streams, showing the vehicle simultaneously from 3 different camera viewpoints. We have considered the simplest visual evaluation involving tracking a single vehicle that appears continuously in the field of view of at least one of the three cameras during the entire experiment duration. Information from the video sequences and the wireless sensors can be used to get information about the timeline of motion events. The tracking algorithm must be able to predict the possible location of vehicle at particular time instants.

5.4 Calculation of 3-D world coordinates

Once the camera internal and external parameters are estimated, the projective transformation matrix can be used to obtain the 3-D world coordinates of the known set of 2-D pixel coordinates representing the centroid of the vehicle. Using the camera internal and external parameters, it is possible to estimate the World coordinates of a point in the image plane. If we have prior knowledge of the scene, it is possible that we can determine the 3D location of a point from a single image. The coordinates of the reference point, 'clock' are (10ft,0ft,4ft) in world coordinate system, we can then assume that the image plane is at z = 1 and the line joining the camera's focal point and the coordinates of the reference point 'clock' in the image plane should intersect the world plane at y = 0. This should give us the world coordinates of the reference point. The equation of the line joining two points A (a,b,c) and X (x,y,z) is given by

$$(a,b,c)+t(x-a,y-b,z-c) \qquad \text{(Equation 17)}$$

Here, the value of't' lies between 0 and 1.

Similarly, the equation of the line joining the camera's focal point and the point in the image plane can be written in terms of (5-21). The line joining the image point (321,45,1) of the reference point and the camera focal point is given by:

$$(a,b,c)+t(321-a,45-b,1-c)$$

Here (a,b,c) is the camera focal point. This equation intersects with the equation y = 0. Solving the above two equations, we get the value of t. The 3D world coordinate of the reference point is

$$(a+(321-a)t, b+(45-b)t, c+(1-c)t)$$

Using the above, the 3-D world coordinates of the reference point, 'the clock' can be calculated with respect to the three cameras and can further be extended to calculate the 3-D world coordinates of the 2-D pixel coordinates representing the vehicle centroid at different time instants in the video sequence. Using initial values of angles and pixel width and height, we were able to map the 2D pixel to 3D world coordinates and vice-versa for the reference point 'clock' for all the three cameras. The world and image coordinates of the reference point obtained are:

World coordinates1 – (9.5ft, 0, 2.7ft)
World coordinates2 – (10.2ft, 0, 2.9ft)
World coordinates3 – (9.7ft, 0, 2.8ft)

Part of the data processing algorithms have been published in [6] at TridentCom 2007. The algorithm to obtain the vehicle location in the world plane from the 2-D pixel coordinates is a two step iterative camera calibration procedure. Each of the cameras is first calibrated using the knowledge of two reference points, the clock and one other reference point common to all the three cameras. Initial guess of the camera rotation angles and principal point are made. Once the three cameras are calibrated separately, an iterative process to refine the camera internal parameters obtained earlier by calibrating the cameras further is developed. The iteration continues till the estimates obtained using the three cameras are very close (by comparing the difference with a preset threshold). The estimation error in the vehicle location estimates obtained using the sensors and the cameras' (using the iterative procedure) is around 0.77ft.

5.5 Camera calibration using non-linear minimization techniques (fminsearch)

We wish to find the camera parameters that accurately predict the pixel coordinates of a point in 3D from the point's world coordinates. As input, we have a reference points that is common to all the three cameras. The 2-D image coordinates which are known to us can be represented as a function of the unknown parameters and the known world and pixel coordinates. Equation 16 can be rewritten in this format which represents the difference (error) between the actual image coordinate of the reference point and the estimate image coordinates of the same reference point:

$$\left[x_{im}-\left(o_x-f_x\frac{r_{11}Xw+r_{12}Yw+r_{13}Zw+Tx}{r_{31}Xw+r_{32}Yw+r_{33}Zw+Tz}\right)\right]$$
$$\left[y_{im}-\left(o_y-f_y\frac{r_{21}Xw+r_{22}Yw+r_{23}Zw+Ty}{r_{31}Xw+r_{32}Yw+r_{33}Zw+Tz}\right)\right] \quad \text{(Equation 18)}$$

We have assumed that the pose and location is known for all the three cameras. We have assumed the optical center to be known. The only unknown parameters are the pixel width and pixel height. We have a total of 2 unknowns. A nonlinear minimization technique is used estimate the unknown parameters in such a way that the sum of square of the difference between the observed images of the reference point and what is predicted by the current estimate of model parameters with respect to the three cameras is minimum. The formulation for the problem can be written as:

$$F=\sum(x_{imgi}-x_{esti})\wedge 2+(y_{imgi}-y_{esti})\wedge 2$$

$$F=\left(\begin{array}{l}\left(321-\left(170-f_x\dfrac{10\cos(21\times pi/180)}{10\sin(-46\times pi/180)\sin(21\times pi/180)+4\cos(-46\times pi/180)-1}\right)\right)\wedge 2\\ +\left(45-\left(90-f_y\dfrac{-10\sin(21\times pi/180)\cos(-46\times pi/180)+4\sin(-46\times pi/180)+3}{10\sin(-46\times pi/180)\sin(21\times pi/180)+4\cos(-46\times pi/180)-1}\right)\right)\wedge 2\\ +\left(128-\left(100-f_x\dfrac{10\cos(88\times\pi/180)+15}{10\sin(-66\times\pi/180)\sin(88\times\pi/180)+4\cos(-66\times\pi/180)-1}\right)\right)\wedge 2\\ +\left(164-\left(170-f_y\dfrac{-10\sin(88\times\pi/180)\cos(-66\times\pi/180)+4\sin(-66\times\pi/180)+3}{10\sin(-66\times\pi/180)\sin(88\times\pi/180)+4\cos(-66\times\pi/180)-1}\right)\right)\wedge 2\\ +\left(78-\left(40-f_x\dfrac{10\cos(83\times\pi/180)+23}{10\sin(-67\times\pi/180)\sin(83\times\pi/180)+4\cos(-67\times\pi/180)-1}\right)\right)\wedge 2\\ +\left(130-\left(135-f_y\dfrac{-10\sin(83\times\pi/180)\cos(-64\times\pi/180)+4\sin(-64\times\pi/180)+3}{10\sin(-64\times\pi/180)\sin(83\times\pi/180)+4\cos(-64\times\pi/180)-1}\right)\right)\wedge 2\end{array}\right) \qquad \text{(Equation 19)}$$

Here, the 6 angles are replaced by the respective values for each camera. The average pixel width and height of all the three cameras (fx = 13.7, fy = 11.4) is taken as the initial value for the NLS. The function *fminsearch* is used to estimate the camera pixel height and width. *Fminsearch*: It is a multidimensional unconstrained nonlinear minimization (Nelder-Mead).

x = fminsearch(fun,x0) starts at x0 and attempts to find a local minimizer x of the function 'fun'. The option 'GradObj' is included and it returns in the value of the gradient of fun at the solution x. The gradient of fun is included in the objective function. The gradient is the partial derivatives of f at the point x. That is, the ith component of g is the partial derivative of f with respect to the ith component of x.

```
x0 = [13.7,11.4];
options = optimset('display','iter','GradObj','on','DerivativeCheck','on') ;
[x,fval,exitflag,output]=fminsearch(@allcamsknownpose,x0,options);
```

For the given initial values, the solution obtained is:

x = (12.97   14.78)

fval = 43.47 (value of the function at solution x)

exitflag =   1 (indicating that the function has converged)

output =

iterations: 47

funcCount: 92

algorithm: 'Nelder-Mead simplex direct search' message: [1x196 char]

The value of the gradient at the solution, x given by

G = 1.0e-004 * (-0.5593   -0.4008)

The gradient of the objective function at the minimum 'x' is almost equal to zero.

Using the estimated values of the pixel width and height, we were able to map the 2D pixel to 3D world coordinates and vice-versa for the reference point 'clock' for all the three cameras. The world and image coordinates of the reference point obtained are:

World coordinates1 – (9.5ft, 0, 2.7ft)

World coordinates2 – (10.1ft, 0, 2.8ft)

World coords3 – (9.5ft, 0, 2.8ft)

Inorder for the image coordinates to match with the actual coordinates, the value of the optical center for camera 1 is moved to (170, 90)

Three independent trajectories with respect to the three cameras are obtained that represent the vehicle's trajectories in the coverage area.

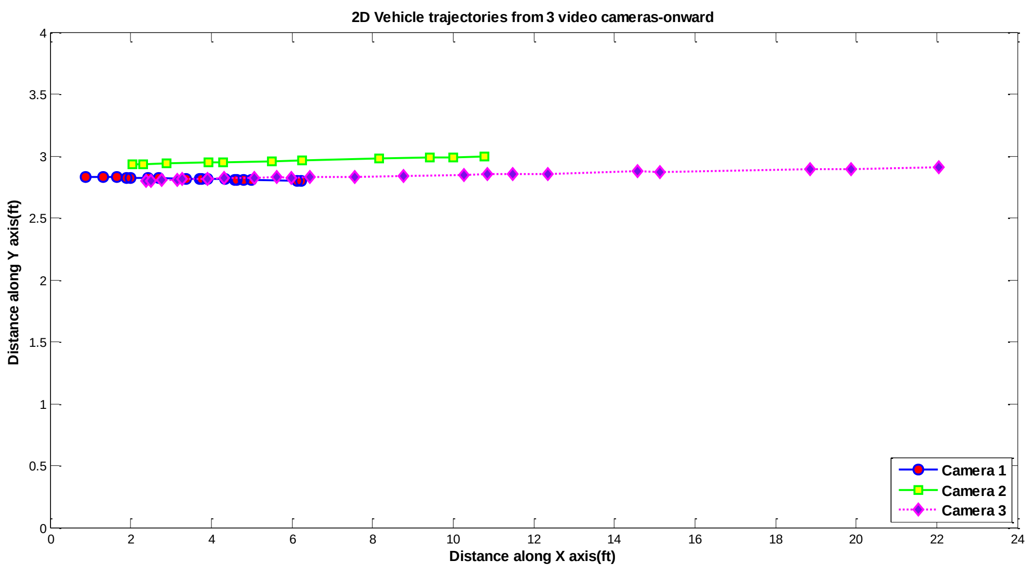

Figure 16: Vehicle trajectory from 3 cameras in the world coordinates - onward direction

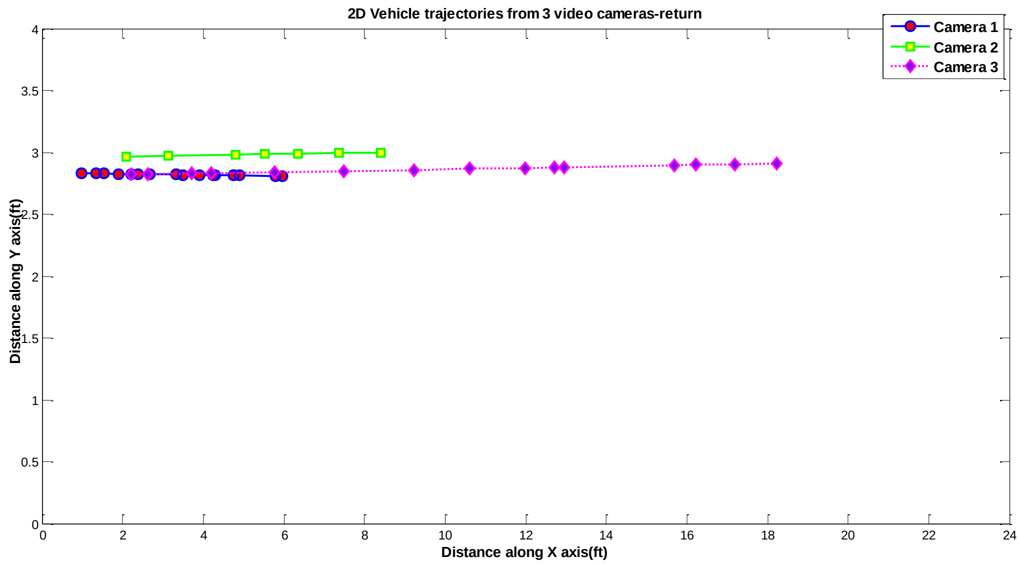

Figure 17: Vehicle trajectory from 3 cameras in the world coordinates - return direction

5.6 Some of the challenges during the experimental study

- Initially in the experiment, we placed all the ten sensors on the same side of the coverage area with the idea that having more number of sensor measurements in the given area would yield more instants when the vehicle's presence is detected. Instead we had many missed detections. On the onward path, almost all the ten light sensors detected the vehicle's presence. But on the return path, the flashlight was facing towards the other side of the coverage area where there were no sensors placed. Not even single sensors detected the vehicle's presence. The acoustic sensors were very expected to detect the vehicle on both the onward and return path. But the acoustic sensor measurements were very unreliable. Hence we only considered light sensors for vehicle detection and placed five sensors on each side of the coverage area.

- Some of the constraints that we faced during the experiment were to maintain the consistency of vehicle motion across cameras, controlling the speed of the vehicle using the remote control was very difficult. We assumed a constant velocity motion model for the vehicle.

- There is a need for designing algorithms that can efficiently fuse the evidence available at each of these cameras into a consistent and robust estimate. Specifically, since we are interested in object detection and tracking, it is important to determine whether an object is present. If an object is present within the field of view, it is also of interest to estimate its pose, appearance, and identity. Robust data fusion methods for surveillance systems using multiple cameras is discussed in [17].

- In typical visual sensing scenarios, the objects of interest are those that are moving. A moving vehicle can be easily detected using the frame difference or the background subtraction method. Once the object is detected in each of the individual cameras, the next task is to track each object using multi-view inputs.

- In the case of multiple cameras, an important challenge is the association of objects across the camera views. For cameras with overlapping fields of views, an important issue is fuse object location estimates. This requires the use of epipolar geometry and triangulation in the most general case.

- In an ideal noise-free condition, the vehicle location estimates arising from each of the cameras should be identical. However, in the presence of errors in camera calibration and vehicle centroid calculations, the location estimates in the world plane are no longer identical. In order to fuse the vehicle location estimates in our experiment we take the arithmetic average of estimated vehicle locations from each of the three cameras, as the trajectories obtained by the three calibrated cameras are very close.

5. 7 Vehicle trajectory from multi-cameras

In the case of multi-camera systems, an important task is the association of the vehicle locations across the camera views. The vehicle location estimates in the 2-D pixel coordinates using the background subtraction method are mapped to corresponding 3-D locations in the world plane using the Self calibration technique.

Each camera has a set of 3-D estimates of the vehicle location. Since the values from three cameras are close but not exactly the same, we need to find a method to fuse these estimates. In order to obtain a single trajectory that could better represent the vehicle's trajectory, we calculate the arithmetic average of the vehicle locations as estimated by the three cameras at every time instant. For example, in the initial few seconds, we have the estimates of vehicle location only from the first camera; the estimated vehicle location with respect to only the first camera is used. The trajectory obtained represents the motion of the vehicle. The trajectory shown in the figure is close to a straight line, which agrees with what we observed.

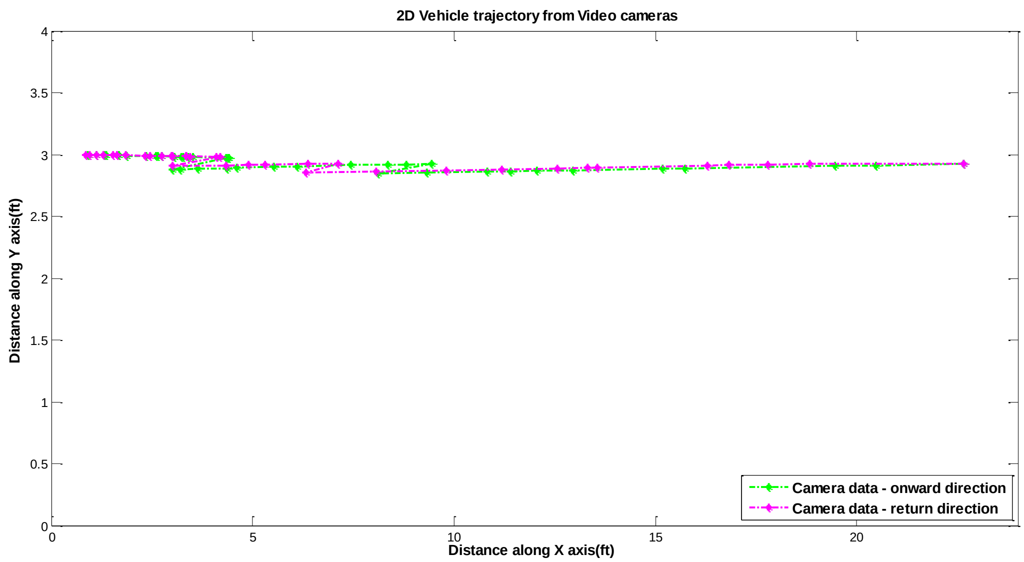

Figure 18: 2-D average vehicle trajectory from video cameras

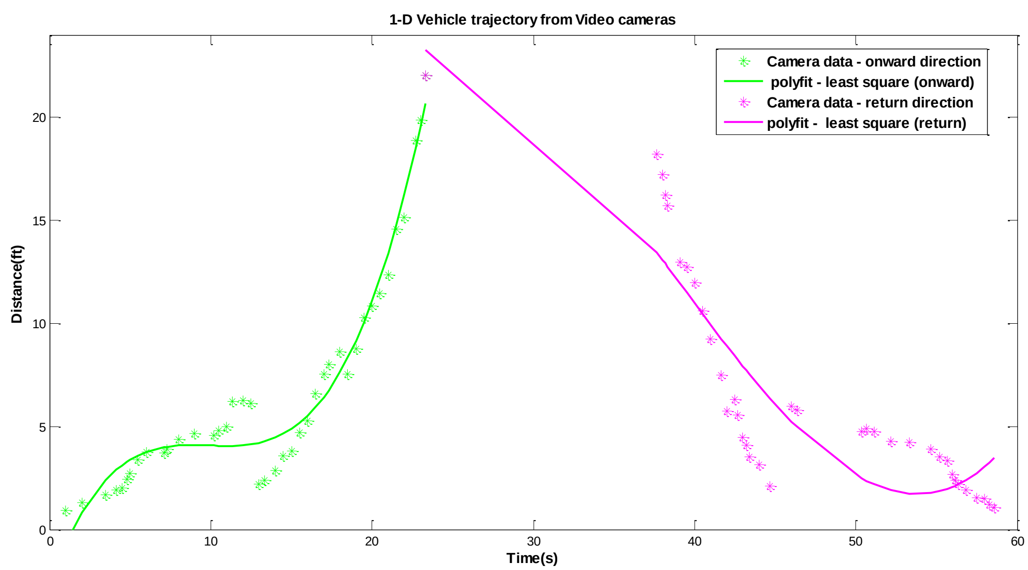

Figure 19: 1-D average vehicle trajectory from video cameras

In Figure 18, both the values of x and y coordinates are plotted and in Figure 19, only the value of the x coordinate is plotted against time.

5. 8 Vehicle trajectory using wireless sensors

The coverage area of the testbed has 10 sensors placed on both sides and approximately 4 feet apart. Significant changes in the light and acoustic sensor readings are observed when the vehicle passed by the sensors correspondingly. Such changes in the sensor readings indicate the time instants (both onward and reverse directions) when the vehicle passes by the sensors. The vehicle locations can be estimated from the locations of the Micaz motes which have a significant increase in sensor readings (mic, light, etc,) corresponding to the vehicle's presence in the vicinity of all the sensors.

Once we have the approximate time instants for both onward and return paths, the next step is to describe the motion of the vehicle, and hence the location of the vehicle, at any particular time instant. With the available distance and time instants, the equation describing the vehicle motion in both onward and reverse directions can be generated by fitting the existing data in the least-squares sense. The curve fitting process fits equations of approximating curves to the raw field data. Nevertheless, for a given set of data, the fitting curves of a given type are generally not unique. Thus, a curve with a minimal deviation from all data points is desired. This best-fitting curve can be obtained by the method of least squares.

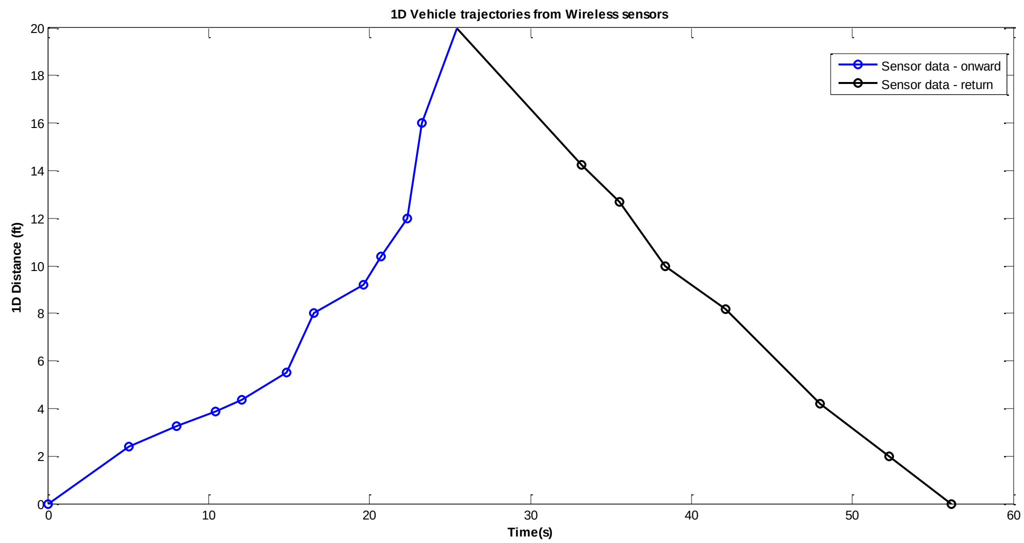

Figure 20: Vehicle trajectory using wireless sensors

Figure 24 represents the sensor-estimated 1-D vehicle locations along x-axis. The trajectory is obtained using the vehicle detections from the sensor measurements.

5. 9 Integration of vehicle detection and tracking

Tracking takes advantage of the fact that it is very unlikely for a vehicle to show up only in one frame. Tracking is considered a process of data association that links the detection results from individual frames to a temporal trajectory. Different sensors may have different sampling rates and can acquire signals in non-regular intervals. In general, surveillance cameras are not synchronized. However, computers' clocks can be synchronized and time stamps can be assigned to frames. This means that we cannot synchronize frames, but we can select frames that belong to time interval of some duration (time tick). The time tick should be big enough to contain at least one frame from each camera but small enough to allow motion only inside the current location or to an adjacent location. In our experiments time tick is equal to one second.

A method for vehicle detection and tracking includes acquiring video data including a plurality of frames, comparing a first frame of the acquired video data against a set of other frames to form hypotheses for vehicle detection/tracking, and verifying the vehicle hypotheses using a set of course-to-fine constraints to detect and tracking the vehicle within one or more subsequent frames of the acquired video data. We compare the video data against the sensor data measurements that are used for vehicle detection. We validate the accuracy the whole process by comparing the vehicle location estimates at certain time instants to the sensor locations that are known prior to the experiment. Video based tracking aims to develop object trajectories over time by using a combination of the object's appearance and movement characteristics. The 2-D object tracks are combined to locate and track objects in a 3-D world coordinate system.

The responses of vehicle detectors (i.e., the sensors) are accumulated over time to obtain the trajectory of the detected vehicle locations. The detected vehicle is tracked in the video frames with the help of perspective projection model and the camera parameters. We estimate the vehicle motion by calculating the centroid of the vehicle in each of the frame, and then find its corresponding location in the real world.

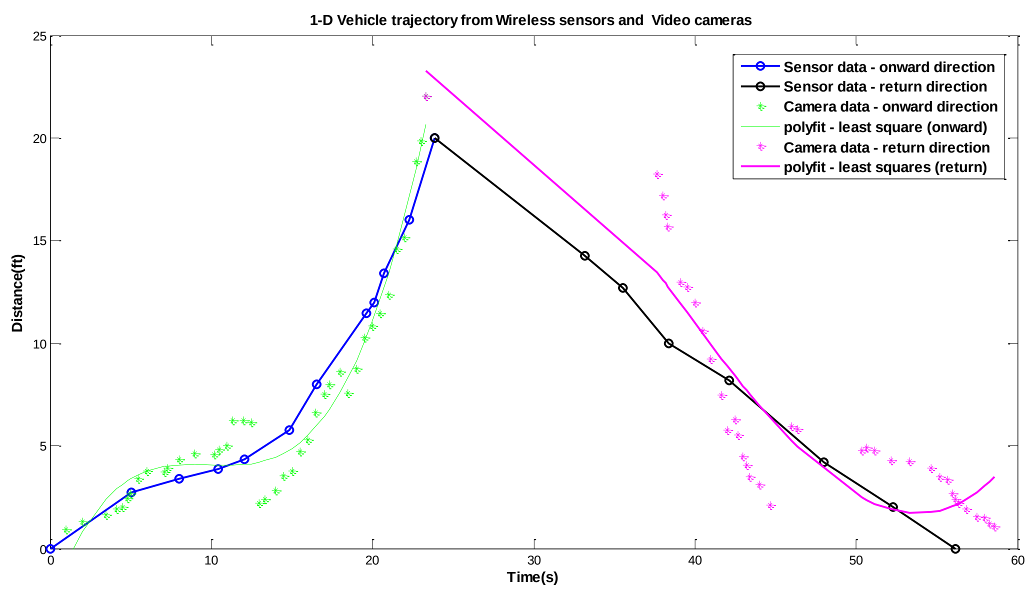

Figure 21: Vehicle trajectory using sensors and cameras

The above figure displays the vehicle trajectory obtained from the video cameras and wireless sensors together.

5. 10 Types of errors in a typical surveillance system

Measuring the performance of smart surveillance systems is a challenging task due to the high degree of effort involved in gathering and annotating the ground truth, vehicle detection and tracking, as well as the challenges involved in defining metrics for performance evaluation. In [31], the typical errors possible in surveillance systems have been categorized. Some of the categories that are applicable to our case are as follows:

- *False Positives:* These are errors that occur when the system falsely detects or recognizes a pattern that does not exist in the scene. For example, a system that is monitoring a secure area may detect motion in the area when there is no physical object but rather a change in the lighting.

- *False Negatives:* These are errors that occur when the system does not detect or recognize a pattern that it is designed to detect. For example, a system monitoring a secure area may fail to detect a person wearing clothes similar to the scene background.

- *Camera related errors:* Errors in the 3-D position of the vehicle due to inaccuracy in the camera calibration data, miscalculation of the vehicle centroid, etc.

The object detection and tracking systems are used to process the test data set and generate results. The objectives of our work are to evaluate the results obtained using the sensor data measurements and video cameras and compare them. We compare the video data against the sensor data measurements that are used for vehicle detection. We could validate the accuracy of the vehicle locations obtained by the video cameras using the sensor data measurements.

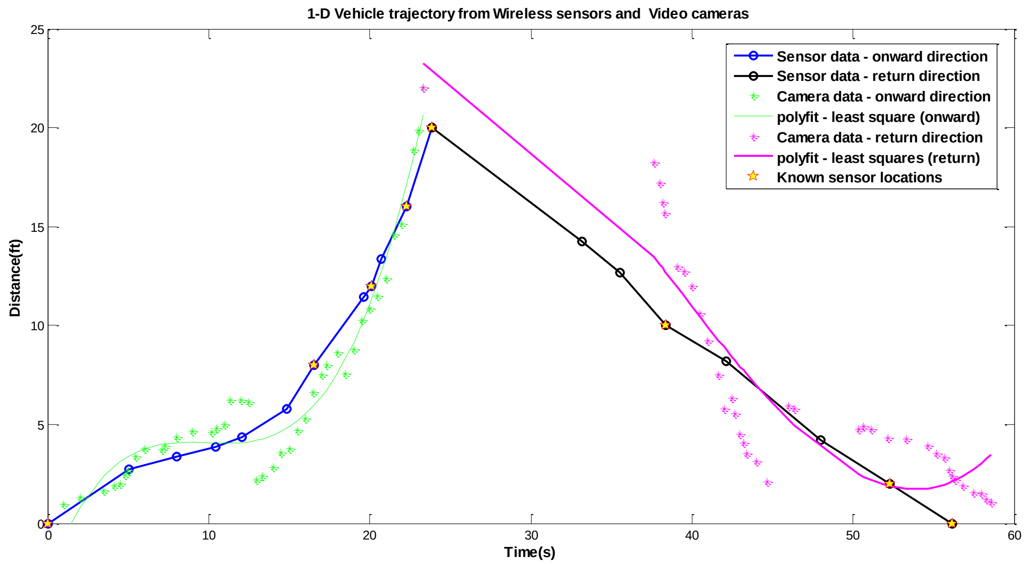

Figure 22: Vehicle trajectories from sensors, cameras and known sensor locations

The above figure displays the 1-D trajectory of the vehicle with respect to both wireless sensors and the video cameras. The sensor locations that are known prior to the experiment are highlighted in the figure. The estimated vehicle locations using the video cameras are compared to the known sensor locations at the particular time instants. The error between the corresponding two vehicle locations is plotted against time. The average estimation error is approximately 0.5 ft.

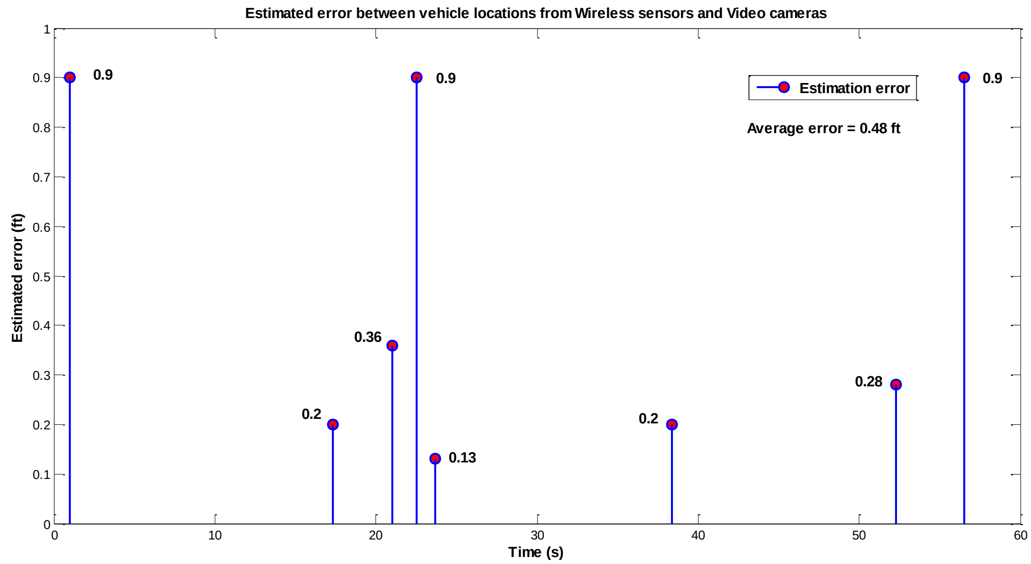

Figure 23: Estimated error between vehicle locations from sensors and cameras

The estimation error between the vehicle locations from wireless sensors and video cameras is calculated. The time instants when the sensors have actually detected the vehicle's presence are considered. We have ten sensors out of which only eight sensors have detected the vehicle's presence. Hence, we have eight time instants. As we can observe from the plot, the estimation error is large during some instants in the experiment. This is mainly due to two reasons. During those time instants, only one camera monitors the coverage area, while at other times, more than one camera monitors the coverage area. Having more than one camera makes use of the redundant information (provided by the videos from different viewpoints), hence more accurately tracking the vehicle than in the case of a single camera. The second reason could be due to calibration errors.

Vehicle detection and tracking can be improved considerably, both in accuracy and time, by taking advantage of the temporal continuity present in the data. We provide a few lines of thinking in Chapter 6.

Chapter 6 - Conclusions and future work

This thesis deals with the development of a testbed with different wired/wireless sensing devices such as the wireless sensors and video cameras. The main motivation is to be able to detect and track a moving vehicle using various algorithms such as sensor data processing, principles of image processing and computer vision. We want to evaluate the performance of the overall procedure by comparing the vehicle trajectories obtained using both wireless sensors and the video cameras.

The data from the light sensors on the Micaz motes is used to detect the presence of the vehicle and to estimate the vehicle location periodically. As the vehicle moves in the surveillance area, approaching the sensor gives a positive peak in the signal. The sensor data that exceeds by a preset threshold, signals an event that the vehicle is detected. Unfortunately, the sensor readings are not always accurate – this is accounted for by the possibility of missed detection and false alarms.

From a technical point of view, the success of an on-road vehicle detection system will depend on the number of correct detections versus the number of false alarms, missed detections. Employing more powerful sensors in vehicle detection applications could influence the system performance considerably. A multi-sensor approach has the potential to yield a higher level of reliability and security. Specific objectives include improving range, sensitivity of the sensors, and incorporating complex computational capabilities. Incorporating different methods of sensor fusion and integration can improve the sensing capacity by using redundant and complementary information from multiple sensors. These sensors will be able to obtain more accurate environmental features that are impossible to perceive with a single sensor.

To tackle the initial stage of the surveillance problem, i.e., the detection of moving targets from a video stream, we have used the frame differencing method to extract the moving target in each frame of the video sequence. This particular method does not take into account the effects of changes in the environmental conditions such as sudden illumination changes. The experiment was conducted in a dark room and we assumed that the background does not vary.

As to future work, the system could be improved on many levels. On the low level it needs more robust background modeling, blob extraction and blob separation techniques. It must be robust against changes in illumination and be able to avoid detecting non-stationary background objects such as moving leaves, rain, snow, and shadows cast by moving objects.

Tracking an object in a video sequence means continuously estimating its location when either the object or the camera is moving. There are a variety of approaches, depending on the type of object, the degrees of freedom of the object and the camera, and the target application. The camera internal parameters are estimated from the set of images by making use of the relationship between the known world co-ordinates and the measured pixel coordinates of a stationary reference point. This information is used to provide the vehicle trajectory for chronologically corresponding frames in the video sequences.

Self-calibration technique was used to estimate the camera parameters. It does not require special objects for calibration. A stationary object which is an alarm clock that appeared in the field of view of the three cameras was used as a reference point. The reference points found in the scene are used to calibrate the camera parameters efficiently. Previous version of the data processing algorithms have been published in [6] at the International Conference for Testbeds and Research Infrastructure for the Development of Networks and Communities, TridentCom 2007. The algorithm to obtain the vehicle location in the world plane from the pixel coordinates is a two step iterative camera calibration procedure. Its estimation error was found to be around 0.77ft.

A non-linear minimization method was also used to estimate the perspective transformation matrix that projects the 3-D world points to 2-D image points. As input, we have a reference points that is common to all the three cameras. The 2-D image coordinates known are represented as a function of the unknown camera parameters and the known world and image coordinates. The objective function simply computes the "pixel error": the difference between the observed images of 3-D reference point and what is predicted by the current estimate of model parameters. The camera parameters estimated in the self-calibration procedure is used as an initial value to the non-linear minimization method. The camera parameters estimated were close to the initial values. Using these parameters, the 3-D world coordinates of the 2-D pixel coordinates of the vehicle centroids are calculated. The average estimation error of the 3-D locations when compared with the location estimated by the wireless sensors is around 0.5ft. With the mentioned challenges, the poor sensitivity of the wireless sensors and limited flexibility in performing the camera calibration, the results are satisfactory. Matlab code and data sets are available for the verification of the experiments.

This thesis report with a simple surveillance scenario involving detecting and tracking a single vehicle that appears and disappears in the field of view from one of the three cameras, without any scene occlusion. Future work may include more challenging scenarios involving tracking several objects, merging and splitting where two or more objects cross paths as observed by the camera and some objects become hidden during the merging.

In typical target surveillance applications, the system needs to detect and track targets simultaneously using the data from multiple sensing. Many target inference problems require both the decision (about the target's presence) and estimation (estimate the track of the target). The testbed can be used to evaluate the joint decision and estimation (JDE) framework. To better understand the connection between the JDE framework and the practical issues, factors involved and the implementation, a clear understanding about the relationship between the requirements in the tradeoff of decision error and estimation error is necessary. In order to achieve this, different JDE scenarios need to be developed using the testbed and meaningful performance evaluation techniques should be developed to evaluate the outputs from various algorithms related to the detection and tracking of the moving vehicle. Different fusion techniques with various practical constraints have to be considered in real target inference problems.

Bibliography

[1] A. Prati, R. Vezzani, L. Benini, E. Farella, and P. Zappi, "An integrated multi-modal sensor network for video surveillance," in *International Multimedia Conference, Proceedings of the third ACM international workshop on Video surveillance & sensor networks*, 2005, pp. 95-102.

[2] J. Ding, S. Y. Cheung, C. W. Tan, and P. Varaiya, "Signal processing of sensor node data for vehicle detection," in *The 7th International IEEE Conference on Intelligent Transportation Systems*, 2004, pp. 70-75.

[3] J. Black, T. Ellis, and P. Rosin, "Multi View Image Surveillance and Tracking," in *Proceedings of Workshop on Motion and Video Computing*, 2002, pp. 169-174.

[4] R. Chellappa, G. Qian, and Q. Zheng, "Vehicle detection and tracking using acoustic and video sensors," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '04)*, 17-21 May 2004, pp. 793-796,Vol3.

[5] R. Cucchiara, et al., "Using a Wireless Sensor Network to Enhance Video Surveillance," *Journal of Ubiquitous Computing and Intelligence (JUCI)*, vol. 1, pp. 1-11, 2006.

[6] M. Yang, H. Chen, X. R. Li, and S. Bandarupalli, "A Surveillance Testbed with Networked Sensors for Integrated Target Inference," in *3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities, TridentCom*, 2007, pp. 1-10.

[7] Crossbow Technology Inc. [Online]. www.xbow.com

[8] Smart Home Toys: 1/3" Sharp color CCD Bullet Camera. [Online]. http://www.smarthometoys.com/bu-581srw.html

[9] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks (Elsevier) Journal*, pp. 393-422, March 2002.

[10] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International workshop on Wireless sensor networks and applications*, 2002, pp. 88-97.

[11] Crossbow Technology Inc: Micaz 2.4GHz Wireless Module. [Online]. http://www.xbow.com/Products/productdetails.aspx?sid=164

[12] Crossbow Technology Inc: MTS-MDA Series Users Manual. [Online]. http://www.xbow.com/Support/Support_pdf_files/MTS-MDA_Series_Users_Manual.pdf

[13] TinyOS Tutorial: Getting started with TinyOS and nesC. [Online]. http://www.tinyos.net/tinyos-1.x/doc/tutorial/lesson1.html

[14] TinyOS Tutorial: Hardware Verification. [Online]. http://www.tinyos.net/tinyos-1.x/doc/tutorial/verifyhw.html

[15] Crossbow Technology Inc: Moteview Users Manual. [Online]. http://www.xbow.com/Support/Support_pdf_files/MoteView_Users_Manual.pdf

[16] Crossbow Technology Inc: Getting started guide. [Online]. http://www.xbow.com/support/support_pdf_files/getting_started_guide.pdf

[17] A. Sankaranarayanan, A. Veeraraghavan, and R. Chellappa, "Object Detection, Tracking and Recognition for Multiple Smart Cameras," in *Proceedings of the IEEE*, Oct 2008, pp.

1606-1624.

[18] N. K. Kanhere, S. T. Birchfield, and W. A. Sarasua, "Vehicle Segmentation and Tracking in the Presence of Occlusions," 2006.

[19] J. Ferryman, A. Worrall, and S. Maybank, "Learning Enhanced 3D Models for Vehicle Tracking," in *British Machine Vision Conference*, 1998, pp. 873-882.

[20] A. Zisserman and R. Hartley, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004.

[21] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, NJ, USA: Prentice Hill, March 1998.

[22] O. Faugeras, Q. T. Luong, and S. Maybank, "Camera Self Calibration: Theory and experiments," in *European Conference on Computer Vision (ECCV)*, 1992, pp. 321-334.

[23] R. Tsai, "A Versatile Camera Calibration technique for high accuracy 3D machine vision using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, vol. 3, pp. 323-344, 1992.

[24] Z. Zhang, "A flexible new technique for Camera Calibration," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330-1334, 2000.

[25] J. Batista, J. Dias, H. Araújo, and A. T. d. Almeida, "Monoplanar Camera Calibration Iterative Multi-Step Approach," in *Proceedings of British Machine Vision Conference*, 1993, pp. 479-488.

[26] L. L. Wang and W. H. Tsai, "Camera Calibration by Vanishing Lines for 3D Computer Vision," in *Transactions on Pattern Analysis and Machine Vision*, vol. 13, 1991, pp. 370-376.

[27] Movie Converter. [Online]. http://www.movietoolbox.com/converter.html

[28] S. C. S. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video," *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 1, pp. 2330-2340, 2005.

[29] R. I. Hartley, "An algorithm for self calibration from several views," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994, pp. 908-912.

[30] D. B. Gennery, "Stereo-camera calibration," in *Proceedings of 10th Image Understanding Workshop*, 1979, pp. 101-108.

[31] L. M. Brown, et al., "Performance Evaluation of Surveillance systems under Varying Conditions," in *Proceedings of IEEE Performance Evaluation of Tracking and Surveillance workshop*, 2005.

Vita

Sowmya Bandarupalli was born in Guntur, India and received her Bachelor's in Technology in Electrical and Electronics Engineering with distinction from Jawaharlal Nehru Technological University, Hyderabad, India in June, 2000. She joined Satyam Computer Services Limited as a software engineer shortly thereafter in July, 2000. During her 3-1/2 years of employment with Satyam Computers, she was involved in the entire Software Development Life Cycle (SDLC) of projects including developing tools related to software engineering. Her technical skills include proficiency in software programming languages like Visual Basic 6.0, ASP, XML, SQL Scripting languages such as VBScript and Javascript.